November 1992 ISSN 0956-9979

VIRUS BULLETIN

THE AUTHORITATIVE INTERNATIONAL PUBLICATION
ON COMPUTER VIRUS PREVENTION,
RECOGNITION AND REMOVAL

CONTENTS

EDITORIAL

If You Can't Stand The Heat...

Few of those in the anti-virus world would deny that the temperature is rising with each passing day. This sapping heatwave within the industry has far-reaching implications, the most serious of these being very basic indeed: how long can developers withstand the heat?

Much of this 'global warming' emanates from the manufacturers themselves. In order to compete with rivals, companies are offering to look after *all* aspects of the virus problem on *all* different platforms. Anti-virus software is now being produced to run under *MS-DOS*, *Windows 3.1*, *VAX/VMS* and even for *OS/2*. To develop and maintain a scanner effectively is not a trivial task. To do the same thing under a number of different operating systems increases the workload enormously. The addition of *NetWare Loadable Modules* to many developers' product lists has fanned the flames still higher.

The virus writers are also adding fuel to the fire. It would appear that they are far more organised than their would-be nemesis. While anti-virus software manufacturers fight amongst themselves to claw their way to the summit of what is an increasingly large heap, the computer underground, oblivious to profit or loss, silently goes about its 'business' - one only has to spend a few hours wandering around the more anarchic Bulletin Boards to find examples of their handiwork.

Viruses are becoming increasingly complex - a prime example of this is seen in this edition of *VB*. The Starship virus is a thoroughly nasty piece of work, incorporating both armouring and polymorphism - it is the first ever virus whose summary box occupies a whole column of *VB*! Disassembling a piece of code like this accurately is an arduous task - this single virus took up many man-hours of valuable time.

In this sweltering atmosphere some companies are sure to seek cooler climes in which to do their business. Mergers and takeovers will become an increasingly common event - there is a fixed market for anti-virus products, and as the recession bites deeper, developers will have no alternative but to tighten their belts.

Already this month, *Symantec* has absorbed one of its competitors in the anti-virus world, *Certus*. *McAfee Associates* has been floated on the stock market, and *Datawatch* has acquired the *Microcom Software Division*. Before the dust has settled from any of these corporate tussles, still more companies are clamouring to become part of the anti-virus gold-rush, each promising that their product will provide the ultimate defence against all computer viruses, old and new. Just like so many of the prospectors who joined the American gold-rush, it seems likely that most will return home penniless, no better off for their troubles.

This string of events occurring within the space of one month should signal something to all but the most green observer: at the very foundations of this industry, things are changing. These business deals are simply the outward signs of stresses which have been building for some time. With a seemingly endless stream of new products offering 'the most comprehensive' protection against computer viruses, this stress will grow. While the anti-virus world is not doomed, events taking place *now* will define the industry's shape in the future.

Competition is normally perceived as a good thing for the end user, as it brings cheaper prices and better products. The problem for the anti-virus software user is how to discover the truth amongst so many volumes of conflicting information concerning each product.

With no universally accepted testing body it is all too easy for users to flock to some self-appointed Svengali who proclaims products to be either clean or unclean on a whim. Snake oil has never been difficult to purchase in the computer industry, and unfulfilled promises of a panacea for all data security ills are still forthcoming.

The competition, which is already cut-throat, is almost certainly going to become even more vicious. Prices will be cut, and over-burdened programming teams will try to squeeze the pips out of their recalcitrant software. How many packages out there already proclaim that they are by far the best product money can buy?

What does all this imply for the user? Well, put simply, beware. 'Bargain basement' software is not necessarily any worse than its competitors, but companies pushing their prices too low cannot survive without costs being cut somewhere. With no centralised body in the industry to keep wolf-like companies at bay, the chances of an ineffective product passing into the hands of users are all too high (and the consequences all too easy to imagine).

The price of anti-virus software may come as an unpleasant surprise to those instigating a data-security policy. Deciding how much to spend and which package to buy is a serious issue, and with so little to guide prospective buyers it is all too easy to be duped. However, for the industry to survive, research costs *must* be met. Those cutting corners may survive by attempting to price others out of the market, but if this occurs, it is the users who will suffer. Money saved now may end up being spent many times over in the future.

NEWS UPDATE

The *ARCV*

Last month, *VB* reported that two new viruses had been uploaded to BBSs around the UK within ZIP-type archive files. Text within the viruses states that they were written by a group calling itself the '*ARCV*' (The Association of Really Cruel Viruses), but at the time of writing, nothing more about this organisation was known. Since then, a lengthy newsletter which claims to have been written by the *ARCV* has been uploaded to bulletin boards in the United States. *Virus Bulletin* has obtained a copy of this newsletter, and the following report is a summary of the information contained within it.

Introduction And Contents

The newsletter consists of eight files, six of which are simply plain text files containing the newsletter proper. The remaining two files are both COM programs which, between them, display a multi-coloured banner and a list of the viruses purportedly produced by the *ARCV*.

The first text file contains the following brief introduction to the group (typographic and grammatical errors courtesy of the *ARCV*):

> Well you may or may not know that we here are one the only Truly English Computer Underground Organisation (And just to piss off the Americans Out there we will spell everything with an 's' not a 'z'). In this and future newsletters we will be dodging Special Branch and New Scotland Yard as we go, as well as putting in the odd virus ASM file, Debug Dump for you all to have fun with. We will also provide information on what's happening (DUDE) out there in Computer Land.

As well as this introduction there is a list of the contents of the newsletter. The edition sent to *VB* contained two hexadecimal dumps of viruses, source code for the Little Brother virus, an application form to join the *ARCV*, a section entitled 'What is The *ARCV*' (sic) and a final 'Closing' section. After this list of contents the following messages are passed on:

> Greetings...To The Guy Who Wrote CHAOS - Thanks Bud
> The Guy Who Wrote FU MANCHU - Are you English?
> Patti 'VSUM' Hoffman - We are here to make your Life HELL!
> John McAfee - To Think if wasn't for us you'd be Unemployed
> The Guy Who Wrote MICHELANGELO - Geta LIFE!!!!!!!!!!!!!
> Terry Pratchett - You Are COOOOOOOOL!
> And Are Carnivorous Plants Really that Boring?

The references to Terry Pratchett occur time and time again in the rest of the newsletter. Terry Pratchett is an extremely well known author of Science Fiction/Fantasy books, and the *ARCV* continually uses short quotes from many of his popular '*Discworld*' books. The *ARCV's* fascination with Science Fiction is also reflected in the name of one of their members: ICE-9 (the name of one of co-authors of the newsletter) refers to a form of ice invented by Kurt Vonnegut in his novel '*Cat's Cradle*'.

Application Form

One of the text files which makes up the newsletter is an application form for *ARCV* membership. This form is filled with pseudo-legal statements and asks a wide range of questions, many of which have absolutely nothing to do with viruses whatsoever.

The rules of the ARCV are littered with self-important, pompous directives governing the conduct of members. A prime example is shown here:

> USE OF DEADLY HACKING FORCE
>
> Except as provided in these sub-sections, No ARCV member shall ever damage delete or in any way tamper with a computer network or system.
>
> Exception 1a-3-1 : Any BBS or system posting or providing Anti-ARCV propaganda may be crashed or deleted.
>
> Exception 1a-3-2 : Any BBS or system posting or providing any ARCV members phone numbers, Password, or personal information may be crashed or deleted.
>
> Exception 1a-3-3 : Any system so approved by the ARCV Council.

The above restrictions would be laughable were it not for the serious nature of the threat. If those systems expressing criticisms of the *ARCV* are 'crashed or deleted' how can they state (see below) that all information should be free?

Perhaps the most intriguing part of the document is the following insight into the motivation behind the *ARCV*:

> APPENDIX A:
> 1. All Information should be FREE!
> 2. Promote Decentralization - Mistrust Authority
> 3. Access to computers should be unlimited and Total
> 4. Hackers should be judged by their hacking ability
> 5. You can create art and beauty on a computer
> 6. Computers can change your life for the better.

Quite how virus writing is intended to promote any of the above points of view is not clear: everything the *ARCV* stands for would seem to provide evidence for the inaccuracy of these statements!

What Is The *ARCV*?

Within the newsletter there is a section entitled 'What is the *ARCV*' which is reproduced in full below:

```
What is The ARCV.

   ARCV is a organisation that is involved in
Writing and Research of computer viruses. We hold
a Library of IBM Computer viruses for the use of
the ARCV members. But as a group we are involved
in viruses for most the main computer types (IBM
PC, AMIGA, ST, MAC). We have a Bi-Monthly
newsletter with the latest virus news from around
the country and from around the world, virus Dis-
Assemblies and other virus Debug Scripts. We have
links with PHALCON/SKISM in the US, we also have
links with some Eastern Europe Virus writers. Are
group is not only limited to virus activities but
other 'Underground' activities also (Hacking,
Phreaking etc.) so any new members who don't write
viruses could be involved in any of the other
activities we are involved in.

   Are members come from the youths of today, at
the moment we are mainly English students that wish
to beat and know more about the system. We come
from a range of backgrounds from the Electronics
side and the Computer side, I myself Apache Warrior
come mainly from the Electronics side but branched
to the Computer side fully around 2 years ago. I
Hack, Phreak and write Viruses, I am the President
of the group (after all I started the group [and also
appear to be the only other member. Ed.]) and I am some what
of an expert on beating the BT phone exchange and
being a BBS A HOLIC that comes in very handy. Now
ICE-9 is also a Electronics guy who turned to the
computer he writes viruses and is into Heavy Metal.
Now the picture put out by the Anti-Virus Authors
is that Virus writers are Sad individuals who wear
Anoraks and go Train Spotting but well they are
sadly mistaken, we are very intelligent, sound
minded, highly trained, and we wouldn't be seen in
an Anorak or near an Anorak even if dead.
```

It *is* true that most anti-virus researchers take (false) comfort from the fact that they imagine virus writers to be socially inept individuals. In fact, many outwardly normal people seem to gain perverse pleasure from writing computer viruses. The newsletter suggests that both members of the *ARCV* show a passionate interest in matters of sartorial elegance, musical appreciation, and social interaction. It appears that they have missed their vocation - perhaps they should be writing for *Vogue* or *Cosmopolitan*?

Viruses

Included within the newsletter are hexadecimal dumps of two viruses: 'Ontario' and 'Sunday'. Both these dumps are in a form which enables them to be read easily into Debug, and no programming skills are needed to produce working virus copies from them. As well as reproducing other people's virus code, the *ARCV* claims to have written the following viruses: 334, 334-2, ALPHA, ARCV93, ARCVXMAS (referred to in this month's update of known viruses as the ICE-9 virus), HIDOS, NICHOLS, REAPER, TMTMID, and ZAPHOD. All these viruses are said to be available on the *ARCV's Virus Library Disk 2*, though this was not included in the software sent to *VB*.

Closing Remarks

In the final section of the newsletter the *ARCV* promises that its publication will be now produced bi-monthly, and previews briefly the next edition.

A great deal of time and effort has gone in to the production of the *ARCV's* first newsletter (time which would have been better spent improving their grasp of the English language). The *ARCV* does not pose a new threat to computer users, as viruses are already freely available to anyone who wishes to spend a short time looking. While their threats should not be ignored, it is likely that the *ARCV* members will soon tire of their fantasy of 'living on the edge.' ❑

Blackmailer Fined

Dr Roy Booth, a lecturer at *Newcastle University,* UK, has been found guilty of blackmail after threatening to unleash a computer virus within a company. In a three day trial at *Newcastle Crown Court* the jury heard that Dr Booth had been hired as a consultant to develop an engineering program by Washington-based *Imec* last May.

Relationships between Dr Booth and *Imec* were soured after Dr Booth ran up a £400 hotel telephone bill during a trip to the United States, which *Imec* said it would deduct from his wages. Dr Booth then threatened to destroy a £200,000 computer program by releasing a computer virus unless his wages claim was settled. The blackmail threat appeared on the screen of a computer the lecturer had returned to *Imec*. It warned that the software he had been developing for one of *Imec's* customers could be destroyed by the virus.

Judge Michael Cartlidge told Dr Booth 'I am not going to impose a prison sentence. You are 27 years old, you are a University lecturer, and you have a wife and child...You have stepped across the line into the field of criminality. You should be thoroughly ashamed of yourself.' Dr Booth was fined £500, and ordered to pay another £500 costs. He must now wait to see whether he can keep his £17,000 per annum post at *Newcastle University*.

After the hearing Dr Booth said 'At the time I felt that I was acting within the law, and obviously deep down I still have that feeling. But with hindsight I would prefer not to have damaged my career for the sake of a 'phone bill.' ❑

TUTORIAL

PC Support Teams - Read This!

When a user finds a virus on his system, the first question is always, 'How do I remove it?'. Later he might want to know where it came from, how it got on to his machine, what sort of damage it does and even how it works - but first he wants it **out**! Removing a parasitic file infector is an easily understood task, but there is some confusion over boot sector virus removal.

Boot Sector Confusion

For the type of virus which attaches its code to an executable file, a general method of disinfection is to identify all infected files, and then, under clean system conditions, delete them and replace them with known clean copies. For viruses which spread by inserting their code into the boot sectors of disks, disinfection is not so simple, and conflicting advice from the instant experts and gurus continues to cause confusion and misunderstanding among affected users. For a precise understanding of the available disinfection methods when dealing with boot sector viruses, it is necessary to have a clear picture of the main boot process and how that is affected by the various infection techniques used by different virus types. Reference to boot sector viruses also includes multi-partite viruses which use boot sector infection methods as part of their replication technique.

Catching The POST

When the machine is switched on, the power supply activates and voltages begin to build towards their normal operating levels. A hardware initialisation timer will eventually trip and kick the main processor into action, whereupon it will start executing a program stored at a fixed address in the Read Only Memory (ROM) of the machine.

This program enables the processor and its associated circuitry to test, configure, recognise and initialise the hardware. In ATs and '386/486 machines, this process collects information from the CMOS area so that the time, date, passwords and peripheral settings can be incorporated into the configuration. At this point the validity of the CMOS is usually checked, and if its checksum has changed the user is prompted to enter the CMOS setup utility held in the ROM. If the contents of the CMOS are valid, the final part of this program completes such tasks as setting up the BIOS addresses in the interrupt table so that incoming software can communicate correctly with the keyboard, disk drives, video monitors and other peripherals. All of the activities invoked by the initialisation sequence from ROM are collectively referred to as the POST (Power On Self Test) routines, and once they are completed the machine is ready to receive some external instructions. The method by which these are collected is part of the design standard and involves asking the hardware to read the very first sector of a disk in the first floppy drive (A:). This attempt to read the floppy disk has three possible outcomes:

- The first sector of the floppy disk is read into memory, and its contents executed.
- A READ ERROR occurs whilst reading the floppy disk.
- There is no disk in the drive, and a DISK I/O ERROR occurs.

If the disk is read successfully, the contents of the first sector are used as new program instructions and the processor will immediately begin to execute them. If a READ ERROR occurs, a counter will be decremented and the processor will try again. Once this counter reaches zero (it usually starts at three), the processor will abandon the attempt to read the floppy disk and either generate a DISK I/O ERROR or display a READ ERROR message and wait for the user to press a key and thus restart the whole disk read routine. If a DISK I/O ERROR occurs, the processor transfers its attention to the first fixed disk on the system and attempts to read its first physical sector.

The original PC design intended that the machine should first try to read the floppy disk, and only if that wasn't available would the fixed disk be accessed. This allows software to be loaded from floppy disk regardless of the state of the fixed disk. Whether the disk is floppy or fixed, the sequence of instructions on the very first sector is referred to as the Primary Boot Code and while there are no limits to the capabilities of this code, it must be remembered that when it executes, there is no operating system available to provide file access services. Processing from this point depends upon the operating system and whether the code is on a floppy or fixed disk.

Floppy Disk Boot Sequence

As far as MS-DOS is concerned, the function of the Primary Boot Code is slightly different on a fixed disk from that on a floppy.

Each time a floppy disk is formatted under MS-DOS a standard sequence of Primary Boot Code is placed on the first sector, which includes details of the disk capacity and layout, two error messages and two filenames (visible as plain text within the sector code). These files may have either COM or SYS extensions and are almost invariably called either BIO and DOS or IBMBIO and IBMDOS

depending upon the source of the original system. The Primary Boot Code is only loaded and executed during the boot process if the disk happens to be in drive A when the machine is booted. When executed, the code refers to the two filenames and checks to see whether they exist on the disk. If they do, they are loaded and will form the initial Disk Operating System. If this code is not present on the disk, an error message similar to the one shown below will be displayed:

```
Non-System disk or disk error
Replace and strike any key when ready
```

If anything else goes wrong with the floppy boot process, the other error message ('Disk Boot failure') will be displayed and again the machine will wait for a key press before attempting the boot process again.

"By far the best insurance against problems from boot sector viruses on fixed disks is to save copies of the MBS and DBS while the machine is clean."

Note that these messages are displayed by the Primary Boot Code and can only occur *after* the boot sector has been loaded and at least partially executed. If an attempt is made to boot from an infected floppy disk which contains no system files, by the time the error message appears, the virus code will be resident and has probably already infected the fixed disk. Switching off the machine will *not* remove the infection and it will be necessary to disinfect the hard disk. Floppy disks contain only one boot sector which is in track 0, head 0, sector 1.

Fixed Disk Boot Sequence

The early versions of MS-DOS used a similar method of booting when a fixed disk was involved. However, as technology advanced and disk capacities became larger, some serious limitations in the way that DOS kept track of available disk space became apparent. This problem was overcome by dividing the disk surface into 'partitions', each of which would appear to the machine as a separate logical disk. An extra level was then added to the boot cycle such that the Primary Boot Code (now called the Master Boot Sector or MBS) locates and loads an additional sector of code (called the DOS Boot Sector or DBS) which then continues as before in its attempt to load the appropriate system files. To locate exactly where on the disk a particular partition begins and ends, a table of track and sector addresses (the Partition Table) is written into the MBS for use by the Primary Boot Code. There are usually four entries in the Partition Table, each containing the addresses of the relevant partition together with an indication of the partition type and a status flag to indicate which partition should be used during the boot process. Only one partition at a time should be flagged as 'active' in this way. To sum up, the fixed disk boot process (for a computer running MS-DOS) is as follows:

BIOS ROM
- ➤ Switch on and execute the POST program from ROM.
- ➤ Load and execute the Master Boot Sector from track 0, head 0, sector 1.

Master Boot Sector
- ➤ Examine the Partition Table and find the entry flagged as active. Load the associated DOS Boot Sector into memory and then execute it.

DOS Boot Sector
- ➤ Load the system files and pass control to them.

Vital Statistics

Before going on to describe how viruses can interfere with this boot process, it should be mentioned that within the DBS there are a number of values which give vital information about the disk. These include the sector size, cluster size, size of the root directory, number of File Allocation Tables (FATs) and the number of tracks and heads. Without this information any subsequently loaded system cannot access the drive properly and the corruption or destruction of these parameters may result in the system refusing to access files on the disk correctly.

Virus Interference

By definition, the routines in ROM cannot be altered and thus the first section of the instruction sequence which can be subverted is the executable code contained in the MBS. This is where most boot sector viruses insert their interception code. The code within the MBS will usually load the remainder of the virus code and relocate it somewhere safe in memory. The BIOS interrupt addresses that the virus requires are then repositioned to point into the virus code so that the virus can interfere with system activity.

Part of this interference may include a redirection routine which supplies a copy of the original MBS whenever the system requests it, thus making the vital parameters available to the system during its initialisation stage, as well as hiding the virus code from scanners. This is why most

anti-virus packages insist on the software being run after a clean boot from a floppy disk.

A small number of boot sector viruses insert their code into the DBS. This has exactly the same effect and is worth noting because of the change in disinfection techniques.

Defences

Since the boot code is loaded before any other software, it is fair to say that no software defence can be 100% effective against all boot infection techniques. Firmware and hardware defences do exist, but they may detract unreasonably from the flexibility of the PC and must therefore be viewed with some caution. Of more concern to users however, is how an existing infection can be removed, and this is where much confusion reigns.

Disinfection Methods:

Sector Save

By far the best insurance against problems from boot sector viruses on fixed disks is to save copies of the MBS and DBS while the machine is clean. There are proprietary programs which will both save and restore these sectors automatically. It is also a relatively simple task to use a disk sector editor (such as *The Norton Utilities*) to save these sectors to files on a floppy disk. In the event of infection the original boot sector can then be restored. This little forethought can save a great deal of trouble.

MS-DOS Formatting

The problem of disinfecting floppy disks is relatively slight - it is usually sufficient to copy all files from the infected disk either to the fixed drive or to a clean floppy, and then reformat the disk using the FORMAT command. The MS-DOS formatting process consists of initialising various groups of sectors to perform their particular function within the MS-DOS disk structure. This will include providing empty File Allocation Tables, an empty root directory area and all the relevant parameters that the system needs for accessing the disk. This process will overwrite the infected sectors and the disk will be restored to health. In the case of fixed disks containing megabytes of data, transferring files is time consuming and in most cases will not be effective.

Imagine a fixed disk partitioned into drives C: and D:. MS-DOS sees drive C: as extending from (say) track 0, head 1, sector 1 to track 500, head 10, sector 17. If a sector contains 512 bytes, this represents a partition size of 47,863,296 bytes (500 tracks x 11 heads x 17 sectors x 512 bytes per sector, *minus* track 0, head 0, all 17 sectors). As far as drive C: is concerned under MS-DOS, the world begins at track 0, side 1 and nothing exists before that! If a FORMAT C: command is issued, only drive C: will be formatted and nothing will be altered on track 0, head 0. This will leave any virus resident in the MBS completely untouched. However, if a virus existed in the DBS (which in our example would be on track 0, head 1, sector 1), the MS-DOS FORMAT command *would* delete it.

Low-level Formatting

This much misunderstood term was much in vogue during the early days of the virus problem. If we were able to see the microscopic magnetic patterns on a disk, we would notice not only the individual sectors containing data bits, but also the areas between sectors where timing, identification and checking information is stored. This inter-sector information is used by the disk controller hardware and is only written when the disk is first initialised.

The process of writing this addressing information is referred to as low-level formatting and most of the early disk controllers contained the code to enable this to be done even when the drive was installed in a machine. The appeal of low-level formatting was that since it applied to the whole of the physical disk, it could be relied upon to destroy *all* information on the disk and was therefore a sure-fire way of deleting virus code. Unfortunately, everything else was deleted too and the disk had to be re-initialised for whatever operating system was required.

Low-level formatting requires intimate knowledge of the drive and sets things such as the interleave factor, write precompensation cylinders and so on. It should only be done by an expert, and then only as a last resort!

The SYS Command

This command was originally designed to enable the hidden files of the operating system to be copied into their correct places on both floppy and fixed disks, thus making a disk into a bootable system disk. Since this command rewrites the DOS Boot Sector, it seems ideal for removing certain types of boot sector virus. As is typical in this industry, confusion has arisen over conflicting advice given on just what the SYS command can be expected to achieve and how differences between different versions affect the boot code.

Early versions of SYS (pre DOS 5.0) copied the two system files onto the disk into highly specific positions. This meant that a disk had to be prepared during formatting (with the /B option) to leave this space available. From DOS 5.0, SYS is capable of moving files around to create the necessary space, so if sufficient space is available, the system can be copied onto any disk.

All versions of SYS will rewrite the DBS on fixed disks or on floppy disks. The version supplied with DOS 5.0 is no different in this respect. The boot code which is written to the disk is contained within the SYS program so it is not possible to mix versions of DOS (i.e. to use a DOS 5.0 SYS program on a DOS 3.3 machine).

Under certain circumstances the SYS command can be extremely useful in removing boot sector viruses, but the process needs to be understood properly.

The system files need space on the target disk. This can only be available if one of the following is true:

- There are no other files on the disk.
- The system already exists and will simply be replaced.
- The floppy was formatted specially with the /B option.
- The PC is using DOS 5, and the disk has sufficient space somewhere on it.

Using SYS to clear DBS viruses from fixed disks should be done as follows:

> Boot from a known clean system disk and copy a clean SYS.COM (or SYS.EXE) file onto it from your original master disk. With the clean boot disk in the A: drive, enter the command 'SYS C:' from the A:> prompt. This will transfer the system files and rewrite the DOS Boot Sector. The contents of the new DOS Boot Sector come from within the SYS program and the current condition of the disk is immaterial.

If you need to disinfect a floppy disk, ensure that your hard disk is free of viruses and boot from it. Place the floppy disk into drive A: and then enter the command SYS A:. With DOS 4.xx and below, this will only work if the disk has been formatted with the /B option, or if it already has the the same version of the operating system on it. With DOS 5, provided that there is sufficient space on the target disk, any boot sector virus will be overwritten and the system will be transferred.

FDISK

A useful program for removing master boot sector viruses from hard disks is the DOS 5 version of FDISK. This program manages the basic partition structure of a fixed disk and enables the user to set the partition sizes that he requires. However, this should not be attempted with versions earlier than DOS 5 since there is a danger of re-initialising the partition and deleting all the data. With DOS 5, under clean conditions, you should enter the command 'FDISK/MBR'.

This will rewrite the Master Boot Sector and remove any virus code in that sector.

All of the above only concerns the Floppy, Master and DOS Boot Sectors. If a virus puts additional code elsewhere on the disk, this will remain untouched. Some viruses put their additional code into clusters marked as bad, while others may damage or corrupt data or the file structure. However, since the boot sector is the point at which the virus enters the system, any further code will remain unexecuted and is a nuisance rather than a threat.

New BIOS Methods

A departure from the original design standard has been introduced by some third party BIOS manufacturers. A flag within the CMOS memory is checked during the boot sequence. The status of this flag determines whether the machine will attempt to boot from the floppy disk or not. This would be an excellent idea were it not for the advent of multi-partite viruses, which will still be capable of infecting the boot sector of the fixed disk [*e.g. the Starship virus, p.15. Ed.*]. It certainly reduces the risk of inadvertently becoming infected with a boot sector virus from an unknown floppy disk.

However, there are associated risks - if either the MBS or the DBS become corrupted and unusable, the machine will be unable to boot from either the fixed or floppy drives. Most of these new BIOSes have a provision for entering the setup program during the POST routines and thus the flag can be reset and the machine rebooted from a clean floppy. There are some BIOSes which have this fixed disk boot feature enabled by a dip switch on the motherboard. In this case the switch needs to be reset to enable the computer to boot from a floppy disk. Either way, setting the option to fixed disk boot is a good idea.

Conclusions

There is a feeling prevalent amongst some users that if a boot sector virus does not seem to be causing any problems, it should be left alone. This short sighted attitude produces what can best be described as a virus broadcast PC. Any write-enabled disks inserted into that machine will become infected and quite possibly transfer the infection to other machines.

It should be clearly understood that there is no such thing as a benign virus. It may not cause problems on *your* machine but inevitably there are other machines which will be severely affected. It is also open to argument that if you knowingly allow a disk containing virus code to be transferred to someone else's machine, you may be committing a criminal act.

✉ LETTERS

Dear Ed,

I should be obliged if you would make it clear to your readers that Jim Bates did not review the authorized edition of *Approaching Zero*, but some severely corrupted version. A version which apparently consists mainly of a cover, an acknowledgements page, a contents list and a bibliography. A version which also contains a collection of pronouncements, statements and opinions that we do not recognise. For example:

1. In our acknowledgements, Jim's name was not listed alongside the members of the computer underworld, the 'sinners' as he likes to call them. His name was included in a quite separate paragraph, among the 'saints'.

2. Rather than making 'only passing reference to the immature, deficient, schizophrenic ... nature of their [the hackers'] personalities', we made absolutely no reference to 'schizophrenia'. We are not qualified to diagnose this disease but, from what we understand, it seems unlikely that anyone suffering from schizophrenia would exhibit hacker-like behaviour. More than anything, hackers are generally single-minded, and obsessed.

3. We did not say that 'Alan Solomon was the researcher called in' to the House of Commons. The phrase used was 'a copy of the virus was sent to Alan Solomon'. Yes, as Jim but very few others know, the story was truncated. But which researcher did what with one particular virus is hardly a matter of world history and it certainly had no place in our book. We tried to interest our readers; not bore them. Alan's insight was that he accurately predicted that the code (which Jim had tried to decipher) was a text message in Bulgarian Cyrillic. Alan also ventured the view that Nomenklatura was written by the Dark Avenger. These opinions were used to illustrate to the layman (to whom the book is addressed) that the virus had probably originated in Bulgaria.

4. We never claimed that the Bulgarians are a 'new master race of computer programmers'. Nor have we ever seen any 'propaganda' to this effect. On the contrary, in tracking down sources as far as possible, we have demonstrated the utter ordinariness of the people involved. Generally, the hackers we have met are charming, intelligent and interesting: nothing like the stereotypes that Jim would clearly have liked us to portray.

5. We simply said 'No one knows how many Lovechild viruses are in existence. Or how many counters are approaching zero'. This is very different from Jim's more colourful 'uncounted numbers are silently counting down all over the world' version.

6. The bibliography was not 'limited to one per author'. Three authors each had two works cited. And if it was 'a strange miscellany', it did reflect the main influences which encouraged us towards further research. It also reflected the range of the book, which would seem strange to Jim, as many areas are clearly outside his range of expertise.

7. Jim may find Burger's book 'odious' but it has been influential, selling over 44,000 copies, despite its 1988 cover price of £17.45 in the UK. And that's not even a hardback.

8. For Jim to dismiss Peter Tippett's work on virus replication as 'discredited' does not expunge it from history. In any event, no one has yet produced a better model and, despite its flaws (discussed in the book), there is evidence to suggest that Tippett's predictions are standing up remarkably well.

Finally, as to the *National Computer Virus Strategy Group*, I don't think that Jim is either in a position to dictate to *New Scotland Yard* or empowered to act as their spokesman. But, if the Group has been or is to be disbanded, then the members should be told - so that any unpaid efforts they are making to help 'the law' can now be channelled in other directions.

Regards,

Bryan Clough
Co-author of 'Approaching Zero'

[*Jim Bates is not the only researcher to call into doubt the veracity of the book 'Approaching Zero' (VB, August 92, p.27). At the Virus Bulletin conference in Edinburgh in September 1992, Vesselin Bontchev publicly questioned the accuracy of certain events portrayed in the book.*

Dr Alan Solomon has also commented on its questionable factual accuracy (see Virus News International, August 1992, pp. 38-39 for his review of the book).

The criticisms voiced in Mr Clough's letter effectively re-emphasise the need for objectivity and accuracy in all things, not least in factual reporting. Ed.]

Dear Sir,

I am writing to correct a number of erroneous impressions given by the news report in *Computer Weekly* that were repeated in the September issue of *Virus Bulletin*. This followed a speech presented by my colleague Brian Jaques at a recent *DECUS* conference.

The first point is a confusion between the number of computer viruses *detected* by *Barclays'* protection mechanisms and those which could be said to have infected our systems. As stated in the article, our procedures require the use of specialist anti-virus software to check for viruses on existing PC systems and on all PC software and data *entering* the *Barclays* organisation from outside. The increasing incidence of viruses being captured and destroyed by our protection mechanisms is therefore indicative of the growing problem of viruses in the external environment, rather than within *Barclays* itself. The external problems would largely not exist if all PC users, (private and commercial) used similar protection mechanisms to ourselves.

Secondly, the article stated that *Barclays* has spent more than £250,000 in the last four years in cleaning computers infected by viruses. This figure is totally false as our protection mechanisms are designed to act before systems are infected. The cost of cleaning which we gave was our worst case experience of £3,000 cleaning costs for a stand-alone PC which had become infected. Our point was to illustrate the costs which can occur if inadequate protection is put in place. Naturally, it does not cost anything like this sum to deal with a virus caught entering the organisation.

This being said, it is worth noting that world-wide *Barclays* spends several hundred thousand pounds per annum in procedures, software and management time to ensure that our systems are protected against virus infection. The external virus threat is real and requires organisations such as ourselves to remain vigilant to protect the interests of our customers.

Finally, thank you for the opportunity to put the record straight. May we request that you agree to publish the points made in this letter, and that the article is amended in your files so that the inaccuracies are not repeated.

Yours sincerely,

Dr P G Dorey
Head Of Information Security
Barclays Group.

[*Dr Dorey's points are noted. Ed.*]

Dear Ed,

KEEP UTILITIES OPEN!

Operating system companies such as *Novell* should remember that systems are best kept open to all add-on suppliers. They can offend no-one, but encourage further developments and popularity of the platform. More sales for all!

By stating that they are to use *Intel's LANProtect* product (although only for internal use), which operates at the *NetWare* 3.11 level - transparently checking commands/files for viruses traversing the network - gives great credibility to the specific product technology, but not the quality. It is the quality of the virus checking that counts! In this case, a *Trend Micro Devices* licence hides behind the Intel marketing name. How good is it? It doesn't appear to have been in the top rated product evaluations. Does it check for unknown viruses?

With this kind of tool, the network supervisor no longer has to rely on user checks. Technology at last provides fully automated background *NetWare* checking. The concept of using VAPs (v2.x) and NLMs (v3.x) is just what users need. But it should be built in by Novell to act as a common driver. Then each user has the ability to decide the brand/quality of virus checker they wish to attach.

The same applies to DOS itself. A generic bios level virus checking device driver needs inbuilding to trap commands/files traversing a PC. Why have we had to wait so long? Similarly major utility suppliers (backup/restore, comms packages etc.) must provide generic hooks into any vendors virus checker. Where there is a conflict of interest (e.g. vendor supplies both a backup/comms product and a virus checker), limiting only their virus checker to the main product will ultimately lose them sales. The users of competitive and better quality products will look elsewhere - we all have limited pockets, and are little concerned with their efforts to lock out competing vendors, purely to gain short-term marketing share at the user's inconvenience and expense.

Again, users prefer choice as to the brand/quality of security product they wish to use on such a critical matter. Users have limited time to check out a mass of separate vendor options/command line switches, when it could all so easily be handled in a standard embedded driver/slot.

Keep things open for greater market potential, and simpler user adaptability.

Yours sincerely,

Mike Kensey
PC/LAN Security Advisor

IBM PC VIRUSES (UPDATE)

Updates and amendments to the *Virus Bulletin Table of Known IBM PC Viruses* as of 23rd October 1992. Each entry consists of the virus' name, its aliases (if any) and the virus type. This is followed by a short description (if available) and a 24-byte hexadecimal search pattern to detect the presence of the virus with a disk utility or preferably a dedicated scanner which contains a user-updatable pattern library.

Type Codes

C = Infects COM files **E** = Infects EXE files **D** = Infects DOS Boot Sector (logical sector 0 on disk)

M = Infects Master Boot Sector (Track 0, Head 0, Sector 1) **N** = Not memory-resident

R = Memory-resident after infection **P** = Companion virus **L** = Link virus

Known Viruses

_150 - CN: A simple virus from Poland, which does nothing but replicate.

```
_150            00B1 04CD 21B8 0242 33C9 33D2 CD21 B440 8D94 FDFF B196 CD21
```

_226 - CN: A 226 byte virus. Awaiting analysis.

```
_226            8BFA 8905 B802 3DCD 218B D88D 94CB 00B4 3FCD 2180 BCCB 0090
```

_491 - CR: A Russian 491 byte virus. Possibly related to the Maffy viruses, in which case it should be named Maffy-491.

```
_491            0510 0005 0300 8ED0 BC20 008C C801 460E E8AF FFFB FF6E 0C3D
```

_1480 - CER: A Russian 1480 byte virus. Awaiting analysis.

```
_1480           80FC 3D74 B680 F44B 75DE 2EFE 06EE 062E 803E EE06 3275 03E9
```

99% - EN: The strange name of this virus is derived from a text message it contains: 'Het 99%-virus heeft toegeslagen.', which translates to 'The 99%-virus has struck.'. The virus may overwrite programs with a small Trojan which displays this message.

```
99%             7416 8A0E 0900 BB36 008A 0732 C1FE C188 0743 81FB 3403 7EF1
```

Adolf - CR: A 475 byte virus. Awaiting analysis.

```
Adolf           80FC 4B74 2380 FC41 7407 E93A 0158 07EB F906 5033 C08E C026
```

Alex-368 - CN: A Russian virus. Awaiting analysis.

```
Alex-368        1ACD 21E8 B600 FC33 C98B D18B D98B F18B F958 071F 6800 01C3
```

Angarsk - CN: A Russian 238 byte virus. Awaiting analysis.

```
Angarsk         B44E 8D56 8890 B93F 00CD 2172 1A8D 96CD 00B8 023D CD21 7209
```

Arriba - CER: A 1590 byte virus. Awaiting analysis.

```
Arriba          AC04 80AA 47E2 F9B9 0100 FC51 B9FF FF90 E2FD 59E2 F62E FF06
```

AT II - CR: Similar to the AT virus family, and possibly by the same author. These four viruses prepend their code to files, instead of appending it. They do not replicate properly, but seem, at least in most cases, to destroy the infected files. Four variants are known, 108, 114, 118 and 122 bytes long, all from Russia.

```
AT II 108       B7B4 400E 1FB1 6CCD B7B4 4006 1F59 CDB7 B43E CDB7 071F 61EA
AT II 114       B8B4 400E 1FB1 70CD B8B4 4006 1F59 CDB8 B43E CDB8 071F 61EA
AT II 118       C5B4 400E 1FB1 74CD C5B4 4006 1F59 CDC5 B43E CDC5 071F 61EA
AT II 122       D6B4 400E 1FB1 78CD D6B4 4006 1F59 CDD6 B43E CDD6 071F 61EA
```

Budo - CER: A Finnish, overwriting virus, which contains two text strings: 'FLOW LIKE A RIVER - STRIKE LIKE A THUNDER' and 'Run time error', the latter being displayed if an infected program is run when the virus is already resident.

```
Budo            BE00 01B9 7A03 BF63 048A 0488 0547 46E2 F852 B440 8B1E 0F01
```

Burger-Twin Peaks - CN: A 1310 byte overwriting virus. It is detected with the Virdem pattern.

Checksum-1569 - CER: This variant is very similar to the other two Checksum viruses, which are 1232 and 1233 bytes long. The increase in size is due to code added to the virus which makes it capable of infecting EXE files.

```
Checksum-1569     832E 0300 6490 832E 0200 6490 8EC0 5657 8BF5 BF00 00B9 2106
```

Clonewar - CN: A small (247 byte) companion virus, which does not seem to do anything interesting other than replicate.

```
Clonewar          8BD8 B9F7 00BA 0001 B440 CD21 B43E CD21 BA1A 01B9 0300 B801
```

Code Zero - CN: This is a 576 byte virus, written by the same author as the VCL. However, it is structurally quite different from the CodeZero virus included in the VCL distribution package.

```
Code Zero         CD21 5EB8 0157 8B4C 168B 5418 CD21 B43E CD21 B801 4332 ED8A
```

Copyright-1205 - CR: Slightly longer than the original Copyright virus, but detected with the same pattern.

Crazy Imp-1402 - CR: This virus claims to be version 1.5 and is probably by the same author as the original one.

```
Crazy Imp-1402    B413 CD2F 33C0 8ED8 832E 1304 068C C88E D848 8EC0 2681 2E03
```

Dark Avenger-Ps!ko-C - CER: This 1459 byte variant is detected with the Dark Avenger pattern.

Dark End - CER: This virus triggers on Oct 15th (or later) every year and attempts to overwrite the first thirty sectors of drive C:.

```
Dark End          520E 1EB8 00E9 CD21 3D34 1274 03E8 2100 585B 5A3B C374 05B8
```

Deicide II - CN: This virus is probably written by the same programmer as the Deicide virus, and, like the Deicide virus, contains childish messages.

```
Deicide II        8A0F 80F1 FF8A D1B4 02CD 2183 C301 81FB 6905 75EC B400 CD16
```

Digger - CEN: A Russian 1475 byte virus. Awaiting analysis.

```
Digger            BB04 0051 B104 2ED2 0159 43E2 F65B 3D9B 1B74 08E9 A8FB B89B
```

Dima - CEN: A Russian 1024 byte virus. Awaiting analysis.

```
Dima              81BE 2E04 4F4D 7445 2BC9 2BD2 B802 42CD 212E 8386 1C04 0772
```

DOShunt - CR: This 483 byte virus overwrites the beginning of files, and places the original code at the end. It activates on June 26th, overwriting the first 128 sectors of the hard disk with random garbage.

```
DOShunt           3D00 4B74 0E3D 00C6 7405 2EFF 2EDF 02B8 B707 CF06 531E 52B9
```

Dr. Qumak II - CR: A 1079 byte virus from Poland, which contains the encrypted message: 'The famous cooperation strikes again: IT IS DOCTOR QUMAK II! Watch out for the next virus from Kraków, Poland!'.

```
Dr. Qumak II      80FC 4B74 0880 FC3D 74E0 E9C2 FE06 5053 5156 5755 1E52 E825
```

Drop - CER: A Russian 1131 byte virus. Awaiting analysis.

```
Drop              80FC 3D74 0F80 FC43 740A 3D00 4B74 5B2E FF2E 3304 568B F246
```

Eastern Digital-1600 - CER: A 1600 byte virus. Awaiting analysis.

```
Eastern D-1600    3D00 4B75 03EB 0F90 3D00 3D75 03EB 0790 9D2E FF2E 7905 5550
```

Enola-2430 - CER: Longer than the original Enola virus, but detected with the same pattern.

F-Soft 458 - CN: This 458 byte Polish virus contains the text '(c) Frodo Soft', but it is in no way related to the Frodo virus.

```
F-Soft 458        32ED 8D97 6B01 53CD 215B 0E07 BA80 00B4 1ACD 21B8 0001 50C3
```

F-Soft 563 - CN: A 563 byte variant of the F-Soft virus. It uses a very short decryption routine, and detecting it with a hexadecimal pattern is not practical.

F-you 593 and **635** - CER: These two viruses are derived from the original 417 byte variant. Unlike the original virus they are capable of infecting EXE files.

```
F-you 593         CD21 2BC1 35FF FF58 7402 FFE0 E974 FFB0 0233 D233 C9B4 42CD
F-you 635         CD21 29C8 35FF FF58 7402 FFE0 E95C FFB0 0231 D231 C9B4 42CD
```

Geek - CER: This 450 byte virus probably comes from the US. It has not been fully analysed.

```
Geek              80FC 4B74 03E9 0C01 5053 5152 1EB8 0043 CD21 5133 C9B8 0143
```

Gyro - CR: An overwriting, 512 byte virus. Easily detected, and unlikely to spread, just like all overwriting viruses.

```
Gyro              01BD 0003 316E 00A1 1F01 3146 004D 81FD 3E01 75F0 FF26 2101
```

Hide and Seek - CN: A 709 byte Japanese virus, which may display the message 'Hi! boy. Do you know 'hide-and-seek' ? Let's play with me!!'

```
Hide and Seek     03F5 B923 0081 34FF FF46 46E2 F8BA 6602 03D5 B946 00BB 0100
```

Highlander - CR: 477 bytes. Awaiting analysis.

```
Highlander        9C80 FCDE 7505 B4ED E90B 0180 FC4B 7403 E903 013C FF75 0532
```

Ice-9 - CR: This 639 byte virus activates in the first week of January of every year, displaying the message 'Happy New Year from the ARCV Released 1 June 1992 Made in England by ICE-9'.

```
Ice-9             3D05 FF74 3F80 FC3D 7405 80FC 4B75 3050 5351 0656 5752 1E2E
```

Ieronim-600 - CR: Closely related to the Ieronim virus reported last month, but slightly longer.

```
Ieronim-600       5B58 0EB8 0001 50CB 80FC 4B75 7206 1653 561E 5250 518B D8B9
```

Ieronim-1581 - CR: A 1581 byte virus. Awaiting analysis.

```
Ieronim-1581      5B58 0EB8 0001 50CB 80FC 4B75 5F06 1653 561E 5250 518B D8B9
```

Infector - CN: A Russian 822 byte virus. Awaiting analysis.

```
Infector          A200 01A0 F302 2EA2 0101 A0F4 022E A202 01B9 0001 BB00 002E
```

Int86 - CR: 500 bytes. Awaiting analysis.

```
Int86             B452 CD21 0653 268B 4714 A380 02FF 3680 0207 26A1 0200 A37A
```

Ionkin - CN: Two Russian viruses, 231 and 2372 bytes long.

```
Ionkin            3F8D 5602 B906 00CD 2173 03EB 6190 8D5E 028B 1F81 FB4D 5A75
```

Itti-Malmsey - CN: An overwriting virus, 495 bytes long.

```
Itti-Malmsey      3DBA 9E00 CD21 93B4 3FB9 0200 BA7F 02CD 2181 3E7F 028B F69C
```

Japanese Christmas-B, C and **D** - CN: Three new variants of this virus, also known as 'Christmas in Japan'. The major differences are in the activation dates and text messages. All three variants are 600 bytes long, like the original, and can be detected with the same pattern.

Keypress-1266, Mubark - CER: The alias 'Mubark' is derived from the string 'Mubark is caw' which is contained within the virus, but it is based on the Keypress virus.

```
Keypr-1266        7405 C707 3401 F9F5 1FC3 F606 1601 0174 0D8C C005 1000 0106
```

Kiss - CER: This virus has limited stealth abilities - it hides file size increases - but does not intercept all read operations. It is 1015 bytes long, but one 1072 byte variant known as 'Apache' also exists.

```
Kiss              743A 80FC 1274 3580 FC1A 7423 80FC 3D74 DD80 FC4B 74D8 80FC
```

Larry, Larry on ▮ Screen - CER: The name of this virus is derived from the string 'Larry on a Screen', which the virus contains. However, it has nothing to do with the 'Larry' computer game series.

```
Larry             8EC0 BF00 028B CFF3 A406 1FFA BF84 008B 05A3 8702 B85E 02AB
```

Lippi - CN: 286 bytes. Contains the string '(C)RomlSoft(LipPI)1991'.

```
Lippi             F646 1501 7407 33C9 B801 43CD 21B8 023D CD21 7224 8BD8 B43F
```

Lyceum - CER: A group of Russian 'stealth' viruses, 1788, 1832 and 1975 bytes long.

```
Lyceum-1832       B4AA F8CA 0200 80FC BB74 F52E 803E EC06 FF74 E780 FC4E 7407
Lyceum-1788       B4AA F8CA 0200 80FC BB74 F52E 803E F006 FF74 E780 FC4E 7407
Lyceum-1975       B4AA CA02 0080 FCBB 74F6 2E80 3E32 08FF 7503 E921 013D 0242
```

Maffy - CN: Two simple viruses, 323 and 478 bytes long, which do not seem to do anything but replicate.

```
Maffy-323         33D2 B800 42CD 21B4 3FB9 0300 8BD5 CD21 7275 81BE 2400 E8FD
Maffy-478         0150 BF00 018B F5A4 AD86 E0AB 03DD 3BDC 7208 58FB FFA6 78FF
```

Matura - CN: This virus is 549 bytes long. It has not been fully analysed, but contains destructive code (INT 26H calls). The string 'MATURA '92' is stored within it in encrypted form.

```
Matura            83E1 1F83 F91E 74DE 3E8A BE39 0380 E701 80FF 0175 0EB4 43B0
```

Meditation - CEN: A simple, 299 byte virus that places itself in front of the files it infects.

```
Meditation        0042 5ACD 2172 1A2E 8B0E 0F01 B440 2BD2 CD21 B801 572E 8B16
```

Mithrandir - PN: A 496 byte companion virus.

```
Mithrandir        BF00 0189 FE83 EEF0 FF06 FC02 B9F0 01F3 A48C C08E D8BA BD02
```

Necros - CR: A 1164 byte polymorphic virus. No search pattern is possible.

No Frills - CER: An 843 byte virus. Awaiting analysis.

```
No Frills         3D32 5475 04B8 0510 CF80 FC4B 7418 80FC 3D74 1380 FC43 740E
```

Nov. 17th-880 - CER: Closely related to the 855 byte variant, which was originally reported as just '855'.

```
Nov. 17th-880     3D75 04A8 0174 1180 FC43 740C 3D00 4B74 0FE9 3802 59E9 0202
```

Nygus - CER: A 757 byte virus. Awaiting analysis.

```
Nygus             488E C08B D826 803E 0000 5A75 EC33 C0AB BF03 0026 8B05 3D40
```

Pixel-748 - CN: A 748 byte virus. Contains slightly variable code at the beginning of the virus. Detected with the Pixel-277 pattern.

Play Tetris - CR: A 522 byte virus. This virus arrived under the name of *Tetris*.

```
Play Tetris       CF86 E0FA 3CCE 7501 CF3C 4B74 0F3C 3D75 03E9 DB00 86E0 FBEA
```

Problem-863 and **Problem-856** - CER: A more advanced version of the Problem virus reported last month.

```
Problem-863       509E 8BE5 8946 0658 E803 005D 9DCF 2E8C 166D 032E 8926 6B03
Problem-856       509E 8BE5 8946 0658 E803 005D 9DCF 2E8C 1666 032E 8926 6403
```

Protect-1196 - CER: Very similar to the Protect-1157 virus and detected with the same pattern.

Raubkopie-Maus - CEN: A 1888 byte virus. Detected with the Raubkopie pattern.

Reklama - CR: A 2723 byte Polish virus, which displays an advertisement, but is otherwise harmless. Infected programs will crash on machines with less than 640K of RAM.

```
Reklama           9D2E FF2E 1000 9C80 FC4B 75F4 3C00 75F0 2E8C 1E14 002E 8C06
```

Tolbuhin - CR: A group of three destructive viruses, 1147, 1004 and 992 bytes long. Probably of Bulgarian origin.

```
Tolbuhin          B42A CD21 80FA 1575 11B8 0903 BA00 00B9 0100 8D1E 0001 CD13
```

Trivial-31B - CN: Another attempt to write the smallest virus possible. This virus overwrites the first file in the current directory.

```
Trivial-31B       3DBA 9E00 CD21 93B4 4049 BA00 01CD 21C3 2A2E 2A00
```

Uruguay - CER: A group of five viruses from Uruguay, with different sizes, but all using polymorphic encryption. The latest variant has some stealth features. No search pattern is possible.

Vbasic-C - CEN: Very similar to the original Vbasic (5120) virus and detected with the same pattern.

Vienna-643, Lydia - CN: Detected with the Vienna-4 and Dr. Q. patterns.

Vienna-719 - CN: Detected with the Violator pattern.

Vienna-849 - CN: Yet another Vienna variant.

```
Vienna-849        ACB9 0080 F2AE B904 00AC AE75 EDE2 FA5E 0789 BCAD 008B FE81
```

Vienna-Violator-C - CN: An 821 byte variant, probably by the same authors as the other Violator variants.

```
Violator C        ACB9 0080 F2AE B904 00AC AE75 EDE2 FA5E 0789 7C79 908B FE83
```

Virdem-Locked and **Virdem-Wonderful** - CN: Two 1336 byte variants very similar to each other and the original Virdem virus, but containing different text messages. Detected with the Virdem pattern.

Yankee-Penza-1210 - CER: Based on some member of the Yankee family, 1210 bytes long.

```
Penza-1210        B440 EB02 B43F E809 0072 023B C1C3 32C0 B442 2E8B 1E31 009C
```

Yankee-2505 - CER: This virus is based on the Yankee virus, but is slightly polymorphic, although it can still be detected with a search string containing wildcards.

```
Yan-2505          83C7 29B9 6F09 B7?? 3025 B7?? 47B3 ??E2 F7B3 ??1F 90E9 08FE
```

Errata:

The pattern published for the V-Sign virus in Jim Bates' article (*VB*, September 92, p.16) is incorrect. The following pattern should be used in preference:

```
V-Sign            1372 FA?? ???? ???? ???? ???? ???? ??CD 1372 EAE9 A601 7698
```

The pattern published for the Palestinian virus (*VB*, July 92, p.3) is incorrect. The following pattern should be used in preference:

```
Palestinian       E872 F2E8 B7FA E8D0 F0E8 08E5 3C01 7535 BFF2 3F1E 57BF 8C1C
```

INDUSTRY WATCH

All Change...

In early October, Californian software giant *Symantec* Corporation, developer of the *Norton Anti-Virus*, announced that it had acquired Ohio based software house *Certus International*, developer of the *NOVI* anti-virus system. Rod Turner of *Symantec* said of the acquisition, 'The *Certus* technology will assist us in the development of the industry's most comprehensive anti-virus solution, and it marks our entry into the systems security market'. A curious remark, given that *Symantec* has purportedly been engaged in the systems security market for some years now.

Meanwhile *Microcom Utility Products Division* based in Durham, North Carolina, which maintains the *Virex* range of anti-virus products for the Macintosh and PC, has been acquired by *Datawatch*, a software and hardware manufacturer based in Wilmington, Massachusetts. Thomas R. Foley, President of *Datawatch* intends to build a 'software operation through acquisition of companies which we believe have promising prospects for growth.' *Datawatch*, or more specifically its wholly owned subsidiary *Personics* specialises in PC software products such as the *Monarch* network package. The company also manufactures PCs to US government TEMPEST standards.

Not to be outdone amidst the tumult, *McAfee Associates* has announced its plans for an initial public offering of 2,100,000 shares of common stock. On August 20th of this year *McAfee Associates* filed a registration document with the Securities and Exchange Commission for a proposed offering of shares at an estimated price of between $13 and $15 per share. If successful, the total flotation of the company will thus raise in the region of thirty million dollars which will undoubtedly help keep the wolf from John McAfee's door.

These seismic developments, while not transforming the commercial battleground, have generated shockwaves which will be felt throughout the industry. The virus war has evidently been highly profitable for a select few and the more successful companies in the field are attractive propositions for acquisition.

One can certainly expect many more mergers and liquidations in 1993 as recession really starts to bite. Just how many of the one hundred plus companies engaged in anti-virus product development will survive entirely unscathed is impossible to tell. What is certain is that a percentage of them will go to the wall, while others will seek refuge under the umbrellas of larger, more diversified companies.

VIRUS ANALYSIS 1

Jim Bates

The Starship Virus

When taking virus code apart, it is my practice to produce, by the end of it, a full printed disassembly which is commented on every line. This is not because I am any sort of perfectionist but simply because if I need to refer to the work later, I invariably find myself unable to remember much more about the virus than its name. As well as the normal disclaimer and limitation messages for the benefit of other researchers to whom these listings are passed, the disassemblies are usually liberally sprinkled with salty observations on the personality, physical attributes and parentage of the virus writer. Of course I have a much freer reign when making these remarks than that allowed within the sober pages of *Virus Bulletin* (maybe one day a virus writer will try to sue me for libel!). However, even my copious store of coarse, cutting and disparaging phrases was stretched when dissecting the latest virus - Starship.

The Starship virus is undoubtedly the most convoluted collection of garbage it has yet been my misfortune to examine. The best that can be said of it is that it is significantly less boring than the usual offerings.

My information is that this is yet another product of the misbegotten group of virus writers in Bulgaria and it certainly contains many of the tricks and devices which are common in their operations. Despite the convolutions however, this virus is quite easy to detect and remove in its boot sector form.

An Overview

The name Starship exists within the encrypted portion of the code and is obviously intended by the writer to be the name of his creation. From a research point of view, this virus has some interesting variations on several themes: code encryption, both static and dynamic, interrupt stripping, code randomisation, mobile routines, armouring, stealth, multipartite - about the only classifications missing are companion and linking!

A simple search pattern is not possible for the parasitic form of the virus, but the initial sector of the boot infection is not encrypted and can easily be recognised. On a clean machine, once the virus has been detected it is also quite easy to remove, and even disks partitioned in a non-standard way can be disinfected, if care is taken.

This specimen is somewhat different from other multipartite viruses in that the same code functions as a boot sector virus (infecting the MBS) on fixed disks, but as a parasitic COM and EXE infector on floppy disks in drive A or B. It should be noted however, that since the parasitic portion infects any COM file greater than 1917 bytes in size, on an infected machine a floppy disk formatted with the install system option set (/S) will certainly have COMMAND.COM infected and possibly the system files too (if they have a COM extension).

This report describes both boot and parasitic installation and infection processes, as well as the more unusual techniques, although the nature of the internal operation rather precludes the usual blow-by-blow description without an abundant collection of diagrams.

Boot Infection And General Operation

As mentioned above, this virus only affects fixed disks and a further limitation is that only disks with an active partition whose type number is below 5 are infected. For reference the relevant type numbers are as follows:

- Type 1 is a DOS 12 bit FAT partition
- Type 2 is a XENIX file system partition
- Type 3 is the obsolete XENIX /usr file system
- Type 4 is the DOS 16 bit FAT partition

This excludes the common DOS Extended partition type and therefore will thankfully limit the platforms on which this virus can exist.

The virus infects the Master Boot Sector of the disk (at track 0, head 0, sector 1) by changing only three bytes. These constitute the progression or Partition start address in the active entry of the Partition Table itself and result in the Master boot code loading the first sector of the virus rather than the Partition Boot Sector. The actual virus code will be located in the last six sectors of the active partition and is in two parts.

The practical upshot of this method of infection is far-reaching. No actual code is changed within the Master Boot Sector; only the size of the active partition is altered. This does mean that some minor work will need to be done to certain scanners to ensure accurate detection of the virus.

The first section of the virus to be loaded is not encrypted and contains code which will load and decrypt the remaining five sectors of viral code and place them into the initial memory locations together with the relevant system insertions. This virus does not 'hook' the interrupts in the usual way via the interrupt table, but inserts new addresses into the DOS function dispatch routine. Thus straight examination of the interrupt table will not reveal tell-tale addresses, although another area of low memory does show positive indications of the virus presence.

The bulk of the code is also mobile, being dynamically encrypted and relocated during machine operation - with special attention being given to this during the operation of TSR programs. Presumably this is done in an attempt to remain difficult to locate. However, as with all of the memory resident viruses that I have examined, these attempts to remain hidden are eventually futile since there must always be some point where the virus code keeps contact with the system services and that remains one of their most vulnerable areas.

In this instance, there is a highly specific memory area at 0000:04B0h which reveals instantly whether the virus is resident. On most machines, this area is reserved for Optical Disk Driver software and will normally contain zeros. The virus checks this area during installation and only becomes resident if it contains zeros. In this case, the virus inserts either an INT 0B0h instruction (0CDh, 0B0h) or two NOPs (90h, 90h). The third byte is a FAR CALL instruction (9Ah).

Self Protected Code

Normally, resident viruses use the DOS services to protect their memory locations in the same way as TSR programs, or else they manipulate the memory control blocks so that the virus code appears to be a legitimate part of the system. The Starship virus occupies around 2.5k bytes of memory and the fact that it makes its own arrangements for protection means that simple memory tests such as that done by the CHKDSK program do not detect any reduction of system RAM.

During the initialisation phase of the infection, the virus uses the single step interrupt to monitor the disk BIOS routine and strip it back to a ROM entry point. This technique (sometimes called tunnelling) is a favourite trick of the Bulgarian virus writers, although it was first demonstrated as one of the hardware features of the Intel 8086 series of processors.

Boot Stealth

Once installed and initialised, the virus monitors system activity in a number of different ways. To avoid the possibility of system malfunction due to the mismatch of parameters within the Partition Table, the virus examines disk access calls and intercepts requests for the Master Boot Sector. These are held while the MBS is read into the caller's buffer and then the three address bytes are replaced with their correct values. The request is then returned with

the corrected MBS. Once again this is a vulnerable point of the virus since use of a simple Partition Table Editor (such as the *Disk Editor* in *The Norton Utilities*) after booting from a clean floppy disk will immediately show that the partition appears to be only 6 sectors long! For example, if the partition table normally shows the active partition starting at track 0, head 1, sector 1 and finishing at track 449, head 6, sector 17 - this will show a partition size of 53,533 sectors. If this machine was infected with the Starship virus, Norton would show the starting address as track 449, head 6, sector 12 and all the other details would be the same - plainly a conflict of values.

System Monitoring

Apart from the detection of programs becoming TSR, other system activity monitored by the code concerns the creation of files. Here, the monitoring routine intercepts system requests to CREATE a file (function 3Ch) by first testing if the file is to be created on either of the floppy drives A or B. It if is, the extension is checked to see whether it is either COM or EXE and if so the name is copied into a buffer maintained by the virus. Only one buffer exists, so a check is made to ensure that it is empty before being used (thus only one file at a time can be marked for infection). The allocated handle is also stored for similar reference along with the creation date and time and the file attributes.

The DOS CLOSE function is also monitored and when the file being created (and referenced by the virus) is closed, the interception routine checks that the file is greater than 1917 bytes and is not already infected. If it meets these requirements, it is infected, then closed and the virus buffer is cleared for the next target.

At first sight this might seem to be an attempt to subvert development machines where program files are continually being created. However, it should be noted that internally, whenever a file is copied, a CREATE file request is issued for the destination filename. Thus on an infected machine, just copying suitable files from the fixed disk to a floppy disk will cause the destination file to become infected. It is important to appreciate that the infection process does not happen when files are copied in the reverse direction (ie: to the fixed disk). Only if a file is being created on a floppy disk, regardless of where it comes from, will it be infected.

Parasitic Encryption

When a file is to be infected, the virus encrypts the whole of its code before writing it to the file. It does this by first making a copy of itself at offset 50h of an available segment (usually in high video memory) and then building a variable and randomised decryption routine into the preceding space (this does not mean however, that the decryption routine is always 50h bytes long). The encrypted virus code preceded by the decryption routine is appended to the target file and appropriate changes are made to the initial bytes (in the case of COM files) or the program header (for EXE files). So when an infected file is run, the virus decryption routine is executed first.

Quite simply, when this virus is invoked from an infected file, it will immediately attempt to infect the Master Boot Sector and active partition of the first fixed disk. If it succeeds, the virus becomes resident and functions exactly as if it had been loaded from an infected boot system.

"No actual code is changed within the Master Boot Sector; only the size of the partition is altered"

File Recognition

Because both the encryption key and the method change with each infection, simple string recognition will not uniquely identify this virus. However, within the virus code there are two distinct recognition techniques used to prevent re-infection of files. I have not been able to check the uniqueness of these methods, so there is a possibility that they might cause false positives if used without any other qualifying conditions. False positives, of course, cause no problems within the virus code, they simply mean that the identified file will not be infected even though it was clean. However, the identification methods are interesting and might prove useful, so I will describe them here:

EXE file infection recognition is achieved by first checking that the header contains the required 'MZ' identifier. The contents of the header are then checked to ensure that the SP field contains a value of 800h - the IP field is 13h or lower - and the result of subtracting the CS field from the SS field is 100h. If these three criteria are met, then the file is deemed already to be infected.

COM file infection uses a completely different routine which checks to see whether the first byte of the program is 0E9h. This signifies a jump instruction and if this byte is found, the virus collects the succeeding word offset and calculates where in the file the destination of the jump is. If this jump is not found, the file is treated as an EXE file.

Once the jump offset has been calculated, the virus reads seventeen bytes from that position in the file and then applies an algorithm which determines whether within those

seventeen bytes there exists a word which represents a call to interrupts 01h, 02h, 03h, 11h, 12h or 13h. Any of these will result in the infection routine aborting since one of them will exist in the entry code to an infected COM file.

Armouring

The writer has expended tremendous energy in an attempt to armour this virus - that is, attempting to make disassembly as difficult as possible by introducing spurious bytes and code instructions intended to trip up automatic disassemblers. The fact that this report has been written is ample evidence that he failed miserably. While I am dissecting virus code I use a number of different monitors, disassemblers and debuggers (both hardware and software) - some commercially available and others that I have developed for my own highly specific purposes. I am delighted to report that even with the heavy armouring in Starship, no modifications were needed to any of my tools in order for them to break down the code accurately into its constituent parts. This is no testament to my ability but rather a measure of the virus writer's programming ability. Any armouring is a challenge; it could slow down the disassembly process considerably - but not in this case!

Trigger

The trigger routine appears similar to a published routine which displays a simulated moving starfield. Typically however, the writer appears not to have understood the algorithm and the routine contains several bugs which result in random garbage being displayed on the screen at random intervals. When this happens, the user should wait until keyboard control returns and then exit the current application as soon as possible (saving any work as necessary). Only video memory is affected and any other corruption can be avoided with care.

Damage

Apart from the deliberately disruptive trigger routine, this virus does not apparently set out to cause deliberate corruption. However, since I contend that there is no such thing as a 'benign' virus, it is relevant to note that when the fixed disk boot infection takes place, the virus makes no attempt to determine whether the disk sectors it uses are already occupied by legitimate files. Thus this virus will cause damage on machines where any of the final six sectors of the partition are currently in use. Conversely, once the virus has occupied these sectors, they are not marked as being in use and they will therefore in due course probably be allocated for use by DOS and be overwritten by legitimate data. This will certainly cause affected machines to crash during the boot sequence.

STARSHIP

Aliases: None known.

Type: Memory-resident Multipartite.

Infection: COM and EXE files longer than 1917 bytes, and Master Boot Sectors.

Recognition:

Files See analysis.

System 90h 90h 9Ah or 0CDh 0B0h 9Ah in 0000:04B0h to 0000:04B2h indicates that the virus is resident.

Hex Pattern for the boot sector pointed to by the address within the active entry of the Partition Table:

```
B937 00BE D606 BFC0 02F3 A4BF
B004 B908 00F3 A41E C506 4C00
```

Intercepts:

INT 13h DISK READ. Cleans MBS and returns.

INT 20h EXIT PROGRAM. Used to indicate to the virus that relocation is necessary as memory allocation will change when an application terminates.

INT 27h TSR. Used to indicate to the virus that some memory manipulation will be necessary.

INT 21h functions:

31h TSR. Re-routed to virus INT 27h handler.

3Ch CREATE FILE. Collects filename if target is floppy disk and extension is COM or EXE.

3Eh CLOSE FILE. Infects if file to be closed was noted by 3Ch intercept.

4Ch EXIT PROGRAM. Re-routed into the virus INT 20h handler.

Trigger: Displays multicoloured garbage to the screen at random intervals.

Removal: Specific and generic disinfection of the MBS is possible. Under clean conditions, identify and replace infected files.

VIRUS ANALYSIS 2

James Beckett

Shattered Glass

Well, we've been waiting for it to happen for some time, and no-one in the trade is very surprised: we have now encountered the first *Windows*-aware virus. Why do I get all the easy jobs?

Proclaiming itself, rather prosaically, 'Virus For *Windows* v1.4', it is only barely describable as *Windows*-aware, but that may well be to its advantage. Hiding no destructive payload bar its own existence, WinVir14 (a convenient name for a lazy typist like me) spreads rapidly within the *Microsoft Windows* environment. When run by any method such as double-clicking on an EXE or PIF file in a File Manager window, an icon in the Program Manager, or issuing a 'Run' command, a whole directory of programs may be infected at once. However, it produces no output on the display, creates no windows of its own and does not interfere with any other windows operations. Thus it remains hidden with no obvious clues to its existence.

Who Needs *Windows*?

The new virus specifically infects *Windows* executables, rearranging the original host code to have itself executed first, as many normal DOS viruses do. However, it uses no *Windows* calls, only knowledge of the New Executable file format. The virus code relies only on standard MS-DOS services being provided.

The operation of *Windows* is itself built upon MS-DOS. *Windows* is not a complete operating system in itself, just an operating environment that takes over the user interface and multi-tasks user programs. Depending on your system capabilities it can give programs protection against disrupting each other (on a good day!), and programs written specially for *Windows* are designed to take account of the added *Windows* facilities (and constraints). Regardless of this, many of the features of DOS are still available to a running program. DOS continues to manage the file system, and *Windows* does just about everything else. Therefore all the virus actually *needs* to know in order to spread is the format of a *Windows* executable file.

Executable programs under standard DOS come in two forms, COM and EXE files. COM files are an old type maintained originally for backwards compatibility (the bane of the computer industry) with the CP/M operating system, and contain just 80n86 code and data, to a limit of 64KBytes. EXE files were introduced to add flexibility and to increase the maximum program size; they contain a special header section with about 40 bytes of system information, plus any amount of relocation information (typically a few bytes for a small program).

With the advent of *Windows*, running as it can in '386 Protected-mode, this is no longer enough, and a much larger header is defined, containing copious amounts of information to control how the program is loaded. Under *Windows*, several instances of a program (say, two copies of WRITE editing different files) may share the same program code in memory, and the header must give *Windows* instructions on how to attempt this without disastrous corruption.

This 'Segmented Executable' or 'New Executable' header is in addition to the normal DOS EXE header, and largely independent. If a *Windows* program is run from DOS, which does not recognise the new header, a stub program is run, which usually prints a message such as 'This program requires *Microsoft Windows*'. Only when run from within *Windows* is the new header examined, and the full application (or, as in this case, the virus) executed.

This header is not well documented - even the *Windows 3.0 Software Developers' Kit*, with a chapter dedicated to *Windows* file formats, was strangely mute on the matter. Fortunately, it *is* described in the *3.1 SDK*, and *The DOS Encyclopedia* (*Microsoft Press*) also has most of the details.

It was with misgiving that I finally placed a call to *Microsoft Technical Support*, but for once I actually managed to navigate the automated 'phone system and reached someone competent who sent me a faxful of useful information - within the hour!

With over ten pages of 8-point printout of just the infected program's *header* information, analysis was then carried out using standard debugging tools, and the virus action pieced together.

Pure And Simple

The author of the virus is certainly a purist - the DOS stub program does not get infected. If an infected program is run from the DOS prompt (even one started from within *Windows*) the virus will not get control and no files will be infected. Only when run directly from *Windows* will anything happen.

The virus uses direct infection - going resident under *Windows* would require familiarity with *Windows'* memory management and clearly the virus authors haven't progressed that far yet. When an infected file is run, every *Windows* EXE file in the current directory will be infected.

An initial section of host code is copied beyond the end of the file, followed by a section of its data. (In protected mode these should be maintained separately). The virus code and data is then written over these two areas.

After infecting all *en prise* files in the current directory, it disinfects the program from which it was activated. This has the (presumably intended) effect that the average user will simply imagine that they mis-clicked on the filename or icon - a very common thing to do - and the next double-click will correctly execute the freshly cleaned program.

This has however another interesting (and useful) side-effect: if the virus-infected executable is run in a directory which contains no other EXE files, it will disinfect itself without infecting anything. As it is non-resident, removal is therefore trivial, as long as the virus can be trusted to disinfect itself perfectly under all circumstances.

The date and time of infected files are maintained, but the Read-Only flag is not masked off before accessing a file.

There are certain complications in *Windows* as regards which files are likely to be infected. Within DOS, the concept of the current directory is a very simple one, and 'CD' determines the directory upon which commands will act. Clicking on an icon in the Program Manager or on a name in the File Manager could result in a PIF file changing the current directory, or a program running from its own directory, or with the directory set to the default \WINDOWS. In the latter case, every *Windows*-supplied program has the chance to become infected (On my test machine, 10 of the 26 *Windows 3.1* executables were infected).

Yet again, it seems that this has been written 'just to show it can be done'; there is no payload or trigger date, the virus exists only to spread.

Origins And Clues

The virus is short and fairly carefully written, making checks on necessary parts of the executable to ascertain infectibility. The routines in it are neatly laid out (certainly making analysis easier) unlike much other virus code which frequently looks very amateurish.

The string 'MK92' is to be found within the data part of the virus, not used as actual data - perhaps initials and the date of writing?

Windows Detection Tools

Cries of vindication are likely to be heard from those who are against scanning under *Windows*, and Joe User may well take them at face value. It is arguable that converting a DOS program into a pretty *Windows* interface has no positive impact on virus detection, and a clean boot and scan from DOS is still sensibly recommended by those developing *Windows* based scanners as being the only secure way to check.

In *Windows* there is going to be much more potential for viruses to hide themselves and subvert the system - this very simple virus is bound to be followed by more sophisticated ones, and anyone creating software to run from *Windows* had better be sure that it cannot be 'stealthed'.

Summary

We were expecting something rather more inspiring from the first *Windows'* virus to hit the streets, but we find that it is simply rolling out the old standard DOS infection methods. However, *VB* has been recieving reports for some time now that pirated copies of the *Windows SDK* are easy to obtain in Bulgaria, and it seems inevitable that Bulgarian virus writers will be exploring this new playground with glee. It is therefore likely that within the next twelve months we will see *Windows* viruses which are capable of taking full advantage of the multitude of new features offered by *Windows*.

The danger is that with several things now happening at once in a user's desktop environment, the extra activity of a virus is even more likely to go unnoticed as it writhes its way through the system.

WinVir14

Alias: None Known.

Type: Non resident, Parasitic.

Infection: EXE files in the new executable format.

Recognition:

Hex Pattern

```
A140 01E8 7201 BAA8 01B9 AE02
90E8 2F01 E852 01BA A801 B9AE
```

System: No recognition in memory as this virus is non-resident.

Removal: Isolate infected file in a directory with no other files in it and execute.

PRODUCT REVIEW 1

Dr. Keith Jackson

Dr Solomon's Anti-Virus Toolkit for Windows

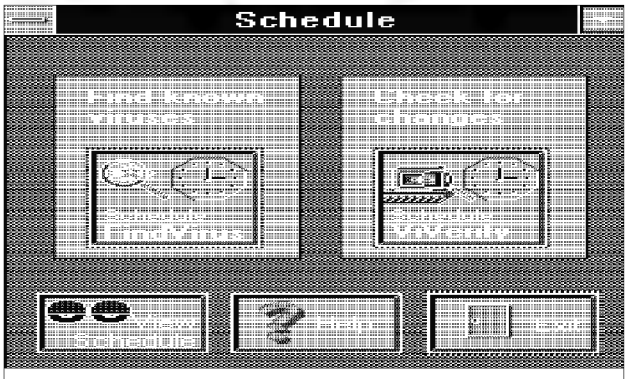
Dr. Solomon's Anti-Virus Toolkit (*AVTK*) was the first product ever reviewed by *Virus Bulletin* (July 1989 pp.13-14), and since that time its progress has been much discussed in the pages of *VB*. Although the *AVTK* has been capable of operating under *Windows* for some time, the release of a *Windows* specific version of the *Toolkit* has only occurred in recent weeks.

The *AVTK* is provided on both 3.5 inch (1.44M for *Windows*, 720K for DOS) and 5.25 inch (1.2 Mbyte for *Windows*, 2x360 Kbyte for DOS) floppy disks, all of which are permanently write protected. A somewhat intriguing assumption seems to have been made by the developers that all *Windows* users have high density drives. Nothing that I can see in the documentation offers lower density disks for penurious souls such as myself.

Installation

To install the *Windows AVTK* simply requires execution of a *Windows* SETUP program. Due to the size of the files, hard disk installation is mandatory. During this process, the DOS version of the *Toolkit* is also installed. The *AVTK* requires 1.62 Mbytes of hard disk space, (of which 620 Kbytes is occupied by the DOS version), and the installation program displays the available space on each hard disk partition to aid selection of the appropriate drive.

After installation is complete, the hard disk is scanned for viruses, and the user is left to press a button to return to the *Windows* Program Manager (after the result of the hard disk scan has been viewed). The final message from the *AVTK* instructs the user 'To create a Rescue Disk exit *Windows* and type RESCUE'. I can't quite see why it is necessary for the user to exit *Windows* to do this, and when I tried it, DOS could not find the RESCUE program (because it was not on the floppy disk last used in drive A:, nor on the DOS path).



Dr Solomon's Anti-virus ToolKit for Windows is one of the best-looking packages ever to be reviewed by *VB*.

Documentation

The *Windows* version of the *AVTK* comes with all of the documentation provided for the DOS version of the *AVTK*, a copy of the 'Virus Encyclopaedia', 9 pages of extra documentation on the *Windows* version of the *AVTK*, and various pieces of bumf. The documentation fits inside a boxed A5 ring binder.

Although the documentation seemed rather sparse I found that the help functions really were rather good, and provided all the information that I needed. The help facilities even contain details of the police departments in various countries that are known to be interested in virus attacks from a crime detection point of view, complete with contact names and telephone numbers - excellent.

Constituent Components

The *Windows* version of the *AVTK* offers almost the same functionality as the DOS version. Indeed during installation of the *Windows* version of the *AVTK*, the DOS programs are also installed, but they do not seem to be called directly by the *Windows* programs: when the DOS version of the virus scanner program is removed from the hard disk, the *Windows* version still operates without complaint. However there are some things that the *Windows* program will not do; for instance attempting to repair a damaged partition under *Windows* simply produces a warning message indicating that only the DOS version of the *AVTK* can achieve this.

All functions are available from drop down menus which can be used to search for known viruses, detect changes in file checksums, repair infected files and irretrievably delete files (multiply overwrite with data). These facilities are available either on a stand-alone PC, or across a network. The most used functions of the *AVTK*: scanning (floppy drives and hard disk drives), checking hard disk file integrity, and examining the Virus Encyclopaedia are also available simply by pushing a button displaying a cute icon indicating its function. In short, even the most naive computer user should be able to navigate their way around this package with ease.

Tools are also provided to inspect files and/or memory (in similar fashion to *The Norton Utilities*), to browse through the encyclopaedia, and schedule automatic invocation of the scanning and checksumming facilities. It was noticeable

that when inspecting memory, the user is warned that this is only possible by executing a DOS program (which it duly does). This illustrates quite succinctly that *Windows* programs are somewhat removed from the low levels at which many (most?) viruses operate.

A paragraph of concise information about each of the viruses known to the *AVTK* (currently 1220 viruses and 940 variants) is provided in the Virus Encyclopaedia. Various methods of locating information about a particular virus are provided, and the software allows multiple copies of the encyclopaedia to be opened. The icon which launches the encyclopaedia changes from a closed book to an open book when execution commences, but the developers may care to note that it gets these two states confused when more than one copy of the encyclopaedia is opened.

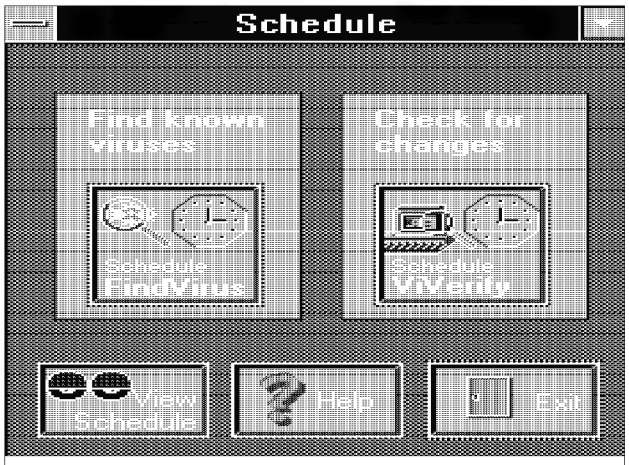
One of the biggest advantages to the *AVTK* for *Windows* is its ability to perform virus scans and integrity checks in the background. The Scheduler allows both of these functions to be performed automatically but is unfortunately limited to only one execution per day. One niggling error I found when using the Scheduler was that if you omit the colon from the middle of (say) 15:32 it takes the entered time as being 15:02 - this needs fixing. After completion of a scheduled run a window pops up to show the user what (if anything) has been found.

Scanning

The *Windows* version of the *AVTK* checked my hard disk in 19.8 seconds (713 files in total [24.8 Mbytes], of which 272 were actually checked). When the DOS version was used this time only dropped to 18.1 seconds, and most of the difference in these timings seemed to be accounted for by longer program loading time (the *Windows* programs are larger). This is very creditable, as most of the *Windows* scanners that I have seen in recent months have been significantly slower than their DOS equivalents. To be honest, nearly all *Windows* programs seem to be slower than their DOS counterparts, and consume much more machine power to provide the same performance: is this *really* progress?

For comparison purposes, when executing under DOS, *SWEEP* from *Sophos* (Version 2.40) in quick scan mode performed the same scan in 18 seconds, and *McAfee's SCAN* program took 26 seconds to perform the same task. All in all, the *AVTK* is still one of the fastest scanners around, even in its *Windows* incarnation.

Dr. Solomon's AVTK has long been shown to one of the best at detecting viruses (In the last comparative review of scanner programs published in *VB* June 1992, p.13, *Dr Solomon's AVTK* attained a perfect score), and given



It is easy to navigate around the package, as the function of each Icon is intuitively obvious

that the functionality of the DOS and *Windows* versions is identical in this respect, I do not intend to occupy space in this review merely confirming this point.

Checksumming

I've written both of the previous *VB* reviews of the *AVTK* and although the reviews were in the main favourable, I did point to weaknesses in the algorithm used to calculate checksums. I'm pleased to see that the user can now select the algorithm to be used: either the *DES* encryption algorithm, a *CCITT CRC* [*A standard developed by the Consultive Committee International Telegraph and Telephone. Ed.*], a 'checksum' (mathematics unspecified), or a simple file size test can be used. These four methods are listed in order of increasing speed of execution, and decreasing level of security/complexity. The calculated checksums are further secured by the user entering a keyword which is used to seed the calculation, which according to the Help system prevents anyone 'reverse engineering your algorithm'. All this is significantly better than previously reviewed versions of the *AVTK*, and lets the user take the decision of whether to trade checksum security for speed of execution. The user can further speed up checksum calculation by specifying the interval between the bytes that are included in the checksum calculation (intervals of 1 to 9 bytes are permitted).

The checksums used by the *AVTK* are extremely easy to calculate and verify, it's merely a matter of using *Windows* buttons to select files, and commence checksum calculation and/or verification. I tested the speed of checksum calculation, using the hard disk described above, including only every ninth byte in the checksum. This took 24 seconds using the *CCITT CRC*, 22 seconds using the unspecified 'checksum' method, 8 seconds when file sizes were

checked, and a whopping 9 minutes 36 seconds when the *DES* algorithm was used (and this was on a 33MHz '486!). The non-cryptographic timings were not significantly altered by including every single byte, but the calculation time using the *DES* algorithm increased to 10 minutes and 15 seconds. As the file read time is the same in all cases, and given the overhead introduced by the *DES* algorithm, I would have expected a far greater difference between the two *DES* execution timings than I actually measured. I suspect that something is awry here: either the interval between bytes is not exactly as stated, or some unstated factor is interfering.

The above figures show that for the non-cryptographic checksums, most of the time is taken up with reading data from files, but when a cryptographic algorithm is used, the speed of execution of the algorithm can dominate. Whether or not a cryptographic algorithm should be used for calculating checksums has been discussed in *VB* on many different occasions, and I do not have the space to summarise the arguments for and against within this review. Regardless, with the *AVTK* it is the user who decides which algorithm is most suitable.

I did find a bug when file size comparisons were in use. Every half dozen executions, the program would stop and display 'Runtime error3 at 0001:1c82', and ask the user to press a button to confirm that the error message had been noted. This is obviously a software bug, but why it should be intermittent I have no idea.

Execution

I have reported very few software bugs in this review, and it therefore seems likely that pre-release testing of the *Windows* version of the *AVTK* has been quite thorough. Every icon is easy to read on a monochrome screen - many's the program I have tested where the icons are almost invisible on my laptop computer. I was pleased to see that the repair facilities correctly detect the capacity of each of my floppy disk drives, but was troubled to find that the Boot Sector repair facility refused to let me repair the boot sector of a 720 Kbyte 3.5 inch floppy disk when it was inserted into a 1.44 Mbyte drive. I can understand the possible problems associated with writing to 360K floppy disks in a 1.2 Mbyte 5.25 inch drive, but these don't occur with a 1.44 Mbyte drive. An oversight perhaps?

Conclusions

I now find myself in the somewhat embarrassing position of having moaned about anti-virus *Windows* programs in the past, pointed out all of the deficiencies in using such programs under *Windows*, but I have just reviewed an anti-virus *Windows* product where I can find no real fault with its implementation, and more to the point it is extremely easy to use. All of the points mentioned above in the review are really no more than quibbles, and no more severe than one would expect from a new product.

If you ask yourself whether *Windows* versions of anti-virus products are necessary, then the answer has to be a firm no. The reasons for this were touched on in the *VB* editorial in last month's issue, with which I agree on the points it makes about security features needing to be based on firm foundations. I won't repeat the arguments from the editorial in this review, suffice it to say that the *Windows AVTK* recommends booting from a DOS disk, and using the DOS scanner before the *Windows* version is used.

The *Windows* version of the *AVTK* does not offer much increased functionality over and above that provided by the DOS version. The changes are mainly confined to the new user interface: it's got more front than Selfridges! No doubt some whizzkid will be able to show that background operation and scheduling are possible under DOS, but this is beside the point - they are much easier to achieve using an operating system which has been designed to carry out such tasks. If the *Windows Toolkit* forces people to use automatically invoked background checksummers rather than just endlessly scanning for known virus patterns then it will have achieved something.

If I was asked whether this is a successful implementation of an anti-virus program, then the answer has to be yes. The amount of development work that has been put into the user interface must have been enormous, and very costly. I expect however that sales may well justify the investment. There are after all a lot of users out there who have totally converted to *Microsoft Windows*, and won't even consider buying software unless it is a true *Windows* executable.

Technical Details

Product: *Dr Solomon's Anti-Virus Toolkit for Windows*

Developer and Vendor: *S&S International Ltd.*, Berkley Court, Mill Street, Berkhampstead, Hertfordshire HP4 2HB.

Tel (0442) 877877, Fax (0442) 877882, BBS (0442) 877883.

Availability: Any PC executing *Microsoft Windows* with at least 2 Mbytes of memory.

Version evaluated: 6.00

Serial number: TK503214

Price: £125 with quarterly updates, £220 with monthly updates.

Hardware used: A 33MHz 486 PC, with one 3.5 inch (1.44M) floppy disk drive, one 5.25 inch (1.2M) floppy disk drive, and a 120 Mbyte hard disk, running under MS-DOS v5.0, *Stacker v2.0* and *Windows 3.1*.

PRODUCT REVIEW 2

Mark Hamilton

PC-EYE - Watching Over Your Computer

PC Enhancements' PC-EYE has come under the microscope on more than one occasion in *Virus Bulletin*. Dr Keith Jackson did the honours last time in October 1991 when he examined version 2.1g - this time I shall look at the company's latest version, 3.0b.

Since I last saw *PC-EYE* (in April 1991), it has undergone a number of changes and enhancements. The first and most visible change is that the A5 manual has been redesigned. It is now a black plastic binder which sports a wraparound cover fastened with two press-studs. The manual is well laid out, with each chapter clearly delineated by card tabs.

The software is supplied on both 5.25 and 3.5-inch disks which have been permanently write-protected, and needs to be installed on a hard drive prior to use. The manual wisely advises scanning the hard drive prior to installation and *PC-EYE's* SCAN program is the only other 'visible' executable on the supplied floppies; the remaining programs are stored in compressed format.

Installation Problems

Having scanned the destination drive, the INSTALL program is run. This prompts the user to supply the name of the directory to which the software will be installed, and which drives he wishes to protect.

The INSTALL program modifies AUTOEXEC.BAT such that it calls a *PC-EYE* batch file (CHECK.BAT) before any other commands are parsed and acted upon.

The installation program forces a reboot of the machine to ensure that changes made to AUTOEXEC.BAT take effect. Unfortunately, this reboot sequence failed on the *Apricot Qi-486* used to prepare this review, and caused the machine to lock up completely. However, as I was forced to power down my computer at this point, the reboot sequence did have the desired effect.

One of the programs executed by CHECK.BAT seemed to modify the PC's operation so that keyboard input is completely disabled - at least on the *Apricot Qi*. By a process of trial and error, I deduced that the problem lay with the BARRIER program, which proved to be incompatible with another TSR which I use on my machine. With this TSR unloaded, I could now continue with the review.

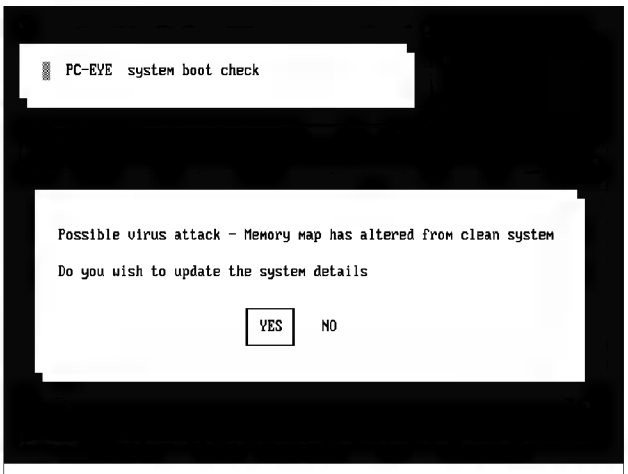
Multi-layered Protection

The developers of *PC-EYE* are obviously well acquainted with the fact that generic checking should form the principal component of any long-term anti-virus strategy. They therefore promote its generic detection as being a better alternative to scanning and herein lies the strength of *PC-EYE*. The generic detection consists of a number of layers.

The first layer is a checksummer which, at install time, reads all the files it deems to be executable - those with extensions BIN, COM, EXE, OV? and SYS (though these extensions can be reconfigured - see below) - and creates a database which contains sufficient information for it to detect changes in these files at a later date. The checksumming method used is a 'dual CRC algorithm'. No other details are given in the documentation.

CHECK.BAT causes FPRINT, *PC Enhancements'* integrity checker, to be executed and this slavishly checks all the executable files on each of the logical drives you nominated at install time. This can add several seconds - or indeed minutes - to the boot-up time, depending on how many files have to be checked and the speed and type of your processor. As an indication, it added 72.9 seconds to the bootup time to test two boot sectors and 519 files (24.5MBytes) on a 25MHz '486DX machine.

The default settings of FPRINT are such that the checksums are only checked once a day. However, this can be easily reconfigured (using the EYE program) to anything ranging from every boot to once every N days. In addition to this, the fingerprint checker is accurate and detected all minor changes I introduced to checked files with reasonable speed (296 Kbytes/second): in this respect, I can't fault it.



PC-EYE takes great care to detect any viruses which become resident at boot-time. Here the package has found a change to the interrupt vectors.

The checksummer notifies the user if any executable files have been added to or removed from the disk. If the checksum database is deleted, the program displays the message 'No checkup file set up'. This is excellent, as warning will be given in the event of anything untoward happening to the checksum database. The checksum creation time is close to the time taken to check the disk's integrity (as it should be) and there appear to be no 'short cuts' in the way the checksummer treats executable files which have different extensions.

Boot Sector Confusion

The checksum database also contains information from the Master Boot Sector and active DOS Boot Sector. PC-EYE, however, falls into the same trap as most other integrity checkers.

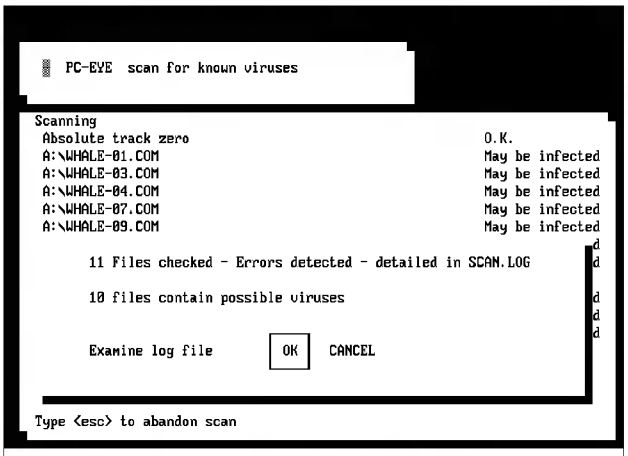
On simple MS-DOS based systems there is no confusion over what information needs to be protected: the Master Boot Sector (MBS) and the DOS Boot Sector. Unfortunately, a disk which has an MS-DOS partition can also have up to three additional partitions, and it is this case which the developers of *PC-EYE* have neglected.

At the end of the MBS of a hard disk is a data area which contains information on the partitions which are present on the disk. In order for the hard disk to be bootable, one of the partitions must also be marked as bootable. *PC-EYE* finds this 'active partition' and saves it in its database.

For the overwhelming majority of PCs this is an acceptable strategy. However, users who wish to use two or more different operating systems on a single fixed disk should be advised that there is a flaw in this approach. In order to switch between these different operating systems at boot-time, a third-party product, like *IBM's Boot Manager* is used. In this case the simple minded approach towards the boot sequence is incorrect, as the sector marked as active within the partition table will not necessarily be the MS-DOS partition. *PC-EYE,* like almost all packages on the market, is unaware of this subtlety, and therefore provides a possible entry point for DOS boot sector viruses.

Memory Checks

Another of the programs executed by CHECK.BAT checks for any changes to the interrupt vectors. This program successfully detected changes I made to the interrupt vectors when I loaded in a TSR program before executing CHECK.BAT. This *should* successfully detect any viruses which change interrupt vectors, but I am not certain that it will successfully detect those viruses which use tunnelling techniques. This said, the vast majority of memory-resident viruses will be picked up by such a check.



Several generations of the Whale virus get washed up.

The Barrier Method

In addition to storing essential information about the various executable files processed by the installation program, *PC-EYE* sets each file's attributes to Read-Only. This provides no protection against viruses - the most simply-written virus can overcome that 'safeguard' without even blinking. However, the BARRIER program uses this setting to provide an extra line of defence.

All calls either to MS-DOS or the BIOS which attempt to change the files from Read-Only are trapped by BARRIER which consequently informs the user that something is awry. This technique is well thought out, and should avoid many false positives, as there are few occasions when write access to an executable is required.

Virus Specific Protection

Possibly the most critical test of *PC-EYE's* capabilities is its ability to detect viruses: however reliable its integrity checker is, if a virus present on the hard disk is not detected on installation then the user is in trouble. Fortunately, the virus-specific part of *PC-EYE* seems to be extremely good. Not only was it capable of detecting all but one of the viruses in the *Virus Bulletin* 'In the Wild' test set, but it successfully identified all of the 'Polymorphic' test set as well. Against the 'Standard' test set, it missed only 3 of the 384 infections, and even when the test set was further expanded, the detection results were very encouraging: when run against my unofficial 'enhanced' test set it scored an impressive 733 out of 784, putting the scanner on a par with some of the best products on the market.

The virus-specific scanner, SCAN, can be invoked either directly, through a batch file (FSCAN) or by the EYE program itself. One of the options given is to execute the

scanner in its 'High Speed' scan mode. Scanning times are then cut down by around 50%, but there is an effect on accuracy: SCAN missed an additional five infections in both of the latter test sets.

The results of each scan are written to a log file, which is overwritten when the next scan is undertaken. This log file is the only way of determining which virus(es) have attacked your system, and the user is given the option to view it if any viruses have been found during a scan. It would be nice to have the option of both saving and printing this file, but as the report is a text file it is easy to undertake any of these operations manually.

Focusing Your EYE...

The default settings of any program are unlikely to suit all users, and having the ability to reconfigure the software easily is of great importance. Fortunately, *PC-EYE* allows all of its many options to be set up using the EYE program itself. This program, which provides on-line instructions and advice on the option selected, allows the user to control the function, security and speed of the constituent programs which make up the package.

The help messages displayed are well thought out and explain how the options you select will affect the system. For example, the BARRIER program can be run in such a way that it will not trap writes to the boot sector of the hard disk. When you select this option you are sensibly warned that you should ensure your boot sector is protected in some other way. This type of on-line advice is an excellent idea, and helps prevent users who have little specialist knowledge of computer viruses from selecting options which are not really suitable for their system.

Conclusion

It is apparent that the developer has expended quite some energy into improving the virus-specific capabilities of its product and this is reflected by it achieving near perfect scores. Moreover, its generic checker performed exactly as it should, and was capable of detecting accurately the changes I made to files.

As a rule, I do not like the idea of trusting the integrity of my disk to either TSR programs or software which is not run directly from a write-protected floppy disk. However, in this case the protection provided seems to cover all of the obvious entry points for a virus. While it is still possible to subvert these measures this should prove difficult to do, as several programs need to be compromised. Therefore, if you are a member of the camp which believes in data protection using TSRs and hard-disk-resident anti-virus software, *PC-EYE* could well be the product for you.

PC-EYE

Scanning Speeds

Hard Disk:

Normal mode (265 Kbytes/sec)	7 mins 58 secs
Turbo mode (149 Kbytes/sec)	3 mins 24 secs

Floppy Disk

Normal mode	15 secs
Turbo mode	7 secs

Scanner Accuracy

VB 'Standard' Test Set[1]		
Normal mode	361/364	99.17%
Turbo mode	356/364	97.80%
'Expanded' Test Set[2]		
Normal mode	733/784	93.49%
Turbo mode	729/784	92.98%
'In The Wild' Test Set[3]	115/116*	99.13%
'Polymorphic' Test Set[4]	150/150*	100.00%

*Tests conducted in Normal mode.

Technical Details

Product: *PC-EYE*

Version: 3.0b

Serial Number: 300014

Author/Supplier: *PC Enhancements Ltd,* The Acorn Suite, 15 Greenleaf House, Darkes Lane, Potters Bar, Herts EN6 1BR.

Telephone: 0707 59016

Fax: Not supplied.

UK Price: £115 + VAT

Test Hardware: All virus scan tests were conducted on an *Apricot Qi486* running at 25MHz and equipped with 16MB RAM and 330MB hard drive. The speed tests were conducted on a *SIR* 486 also running at 25MHz and equipped with 8MB RAM and a CD-ROM drive. *PC-EYE's* scanning speed was tested against a CD-ROM containing 6,483 files (126,814,940 bytes) of which 546 were executable (30,390,671 bytes) and the average file size was 55,660 bytes. The floppy disk test was the same *Microsoft C v5.1 Installation Disk* used in previous reviews.

For details of the test sets used, please refer to:

[1] Standard Test Set: *Virus Bulletin* - May 1992 (p.23).

[2] This unofficial test set comprises 784 unique infections.

[3] *In the Wild* test set: *Virus Bulletin* - June 1992 (p.16).

[4] *Polymorphic* test set: *Virus Bulletin* - June 1992 (p.16).

REVIEWS

PC Plus - The Virus Video

'A must for all serious PC users' runs the blurb on the packaging of this video guide to computer viruses, which claims to 'cut through the hype and break down the myths'. To educate these 'serious PC users' (and, presumably, those with a sense of humour also), a cast of 'leading virus experts' and industry figures are wheeled on to the stage, dusted down, and given their five minutes of fame.

Running time, 50 minutes - not far short of a feature film. Could the director ('Wilf Hey of *PC Plus* fame') really hold the viewers' attention for so long? Well yes, almost - and that is said by a reviewer who is heartily sick and tired of computer viruses.

Such reassuring luminaries as Alan Solomon and Simon Shepherd guide the viewer through the pathology of the beasts themselves, supported by specialists from *Microsoft*, *IBM*, *Lotus* and the *National Computing Centre* to name but a few of the participating organisations. Safe computing practice is introduced gently but effectively as the film progresses. Even *VB's* editor gets a look in (wearing what looks like a toupee), spreading his usual message of doom and gloom, not so much to the aforementioned 'serious PC user', but to the anti-virus software developers - scanners are finished! The end is nigh! You're all doomed! Etcetera.

A software 'walk-through' features offerings from *McAfee*, *Central Point* and *S&S* demonstrated in technicolour. There is far too much *Windows* emphasis in this section for it to be taken seriously by a computer security aficionado. However, Mr Joe Public will presumably benefit from every double mouse click - whoever said that this magazine wasn't elitist? There is also a surreal moment (a dream sequence possibly) as an anonymous 'victim' tells his tale of woe, his face veiled in silhouette, his voiced distorted electronically - if this doesn't make you cringe with embarrassment nothing will.

The film is ambitious and attempts (by and large successfully) to cover most aspects of the virus threat. One possible criticism is that the level of detail is at times too great. Knowledge is assumed on the part of the viewer which confirms the assertion that the video is more suitable for the enthusiast than the casual PC user. There are also one or two weird statistics, the average cost of a virus attack, for instance, is cited at £12,000 - Yikes!

Value for money? With a recommended retail price of £19.99 the answer must be a resounding 'yes'.

Computer Viruses And Anti-Virus Warfare Second Revised Edition

The author of this book is Dr Jan Hruska, notorious for (among other things) his assertion at the IFIP security conference in Brighton in 1990 that the authors of memory-resident anti-virus software were either 'ignorant' or 'unscrupulous'. This seemingly innocuous statement caused a furore amongst a handful of people who perhaps felt they fitted one or both descriptions, and was subsequently censored from the conference proceedings. Fortunately, no such censorship is allowed to encroach on the contents of his book, which is a remarkably straightforward and accurate study of the PC virus threat.

VB's initial review of the book (*VB*, May 1990, p.19) concluded 'the strength of *Computer Viruses And Anti-Virus Warfare* is that it is logical in the way it addresses the subject, clear in its explanations and devoid of the sloppy mistakes which have undermined similar works.' The second edition, which has expanded from 128 to 224 printed pages, continues in this tradition, with updated information about a host of new viruses and recent replication, encryption and stealth mechanisms.

The author has taken pains to document the latest developments in such a fast-moving area of study. Self-modifying encryption, linking, multipartite and companion viruses and the more sophisticated stealth examples are explained, while the proliferation of Virus Exchange Bulletin Boards and the attempted subversion of anti-virus software are discussed. An entire chapter is devoted to *Novell* network protection, a virus hunter's checklist has been introduced, and a section entitled 'virus facts and fiction' destroys some of the more ridiculous myths currently doing the rounds.

At a time when other virus 'experts' are publishing abridged software manuals crudely disguised as 'books', it is refreshing to see a genuine book on this subject which is both easy to read and accurate. *Computer Viruses and Anti-Virus Warfare* is also a ruthless expose of bunkum and half-baked ideas. It is guaranteed to irritate the industry, and for this reason alone must be given top marks - after all, who else would describe the producers of anti-virus products (including himself) as 'the great unwashed'?

Computer Viruses And Anti-Virus Warfare - 2nd Edition

ISBN: 0-13-036377-4

Author: Jan Hruska

Price: £19.95

Available from bookshops or direct from: Ellis Horwood Ltd, Market Cross House, Cooper Street, Chichester, W Sussex, PO19 1EB, England.

END-NOTES AND NEWS

Virus Bulletin* has issued a call for papers for its *Third International Virus Bulletin Conference, which will be held in Amsterdam on 9th -10th September 1993. Papers will be selected for their originality and appeal. Copies of the Conference Proceedings from *VB '92*, priced at £50 plus p+p, are still available from *Virus Bulletin*. For further information contact Victoria Lammer. Tel 0235 555139.

Total Control* has announced the release of *VIS Anti-Virus Utilities Version 4. The package is capable of being run either as a full *Windows* executable or under MS-DOS. *VB* believes that the product's new packaging will sport a completely uncensored full-colour picture of the 'internationally recognised and respected virus researcher' Jim Bates. Perhaps 'Disgusted of Dorking' will find this less distressing than *Total Control's* recent advertisement, which was withdrawn following prudish complaints to the Advertising Standards Authority. For further information contact *Total Control*. Tel 0488 685299.

A video addressing all aspects of Computer Security has been produced as a joint venture between *Barclays Bank, Digital, European Security Forum, Sophos* and *Zergo*. The video, '*Computer Security - Who's Solving The Problem?*', is available from *Positive Image,* UK, price £49.95. Tel 071 407 0625.

The latest release of ***Central Point's* popular *PC-Tools* package now contains a copy of the *Central Point Anti-Virus***, 'the industry's most comprehensive award-winning anti-virus utility'. [*All products are comprehensive, but some products are more comprehensive than others. Ed.*] For further information contact Diane Paternoster. Tel 081 848 1414.

Disknet, Reflex Magnetics'* anti-virus product, now has a sister product, *Disknet For Windows, which runs under 'an operating system now adopted by a majority of corporate users.' The program is capable of informing the user of virus activity by displaying a windowed message. For further information contact *Reflex Magnetics*. Tel 071 372 6666.

The latest edition of 2600, **the *Hacker Quarterly*, contains source code for a primitive computer virus**. The virus, written in 'C', overwrites files, and is capable of infecting all EXE files in directories off the C: drive root directory.

Coinciding with the spate of recent NLM releases, ***Central Point* has announced the launch of its NLM, *Central Point Anti-Virus For Netware***. *CPAV for Netware*, which costs £699 + VAT for a single server licence, claims to provide automated virus detection for *Novell Networks*. Tel 081 848 1414.

***Western Digital* has announced its entry into the anti-virus market** by its launch of a hardware device which takes advantage of the System Management Interrupt feature of power-managed '386/486 processors. Writes to the hard disk are trapped at a hardware level, thus protecting vital areas of the hard disk. Shipping should start in December 1992. For further information contact *Western Digital,* US. Tel (714) 932 6250.

Sophos UK is holding hands-on **Virus Workshops** in Oxford in November (10th-11th), January (26th-27th) and March (24th-25th). Tel 0235 559933.

VIRUS BULLETIN

Subscription price for 1 year (12 issues) including first-class/airmail delivery:

UK £195, Europe £225, International £245 (US$395)

Editorial enquiries, subscription enquiries, orders and payments:

Virus Bulletin Ltd, 21 The Quadrant, Abingdon Science Park, Abingdon, OX14 3YS, England

Tel (0235) 555139, International Tel (+44) 235 555139
Fax (0235) 559935, International Fax (+44) 235 559935

US subscriptions only:

June Jordan, *Virus Bulletin*, 590 Danbury Road, Ridgefield, CT 06877, USA

Tel 203 431 8720, Fax 203 431 8165